ASME Journal of Computational and Nonlinear Dynamics
Online journal at:
https://asmedigitalcollection.asme.org/computationalnonlinear

Yash Vats
The University of Queensland—Indian Institute of Technology Delhi Academy of Research (UQIDAR),
Indian Institute of Technology Delhi,
New Delhi 110016, India;
Department of Mathematics,
Indian Institute of Technology Delhi,
New Delhi 110016, India;
School of Mathematics and Physics,
The University of Queensland,
St Lucia, Queensland 4072, Australia
e-mail: yashvats425@gmail.com

Mani Mehra[1]
Department of Mathematics,
Indian Institute of Technology Delhi,
New Delhi 110016, India
e-mail: mmehra@maths.iitd.ac.in

Dietmar Oelz
School of Mathematics and Physics,
The University of Queensland,
St Lucia, Queensland 4072, Australia
e-mail: d.oelz@uq.edu.au

Abhishek Kumar Singh
Department of Mathematics,
Indian Institute of Technology Delhi,
New Delhi 110016, India
e-mail: assinghabhi@gmail.com

A New Perspective for Scientific Modelling: Sparse Reconstruction-Based Approach for Learning Time-Space Fractional Differential Equations

This paper studies a sparse reconstruction-based approach to learn time–space fractional differential equations (FDEs), i.e., to identify parameter values and particularly the order of the fractional derivatives. The approach uses a generalized Taylor series expansion to generate, in every iteration, a feature matrix, which is used to learn the fractional orders of both, temporal and spatial derivatives by minimizing the least absolute shrinkage and selection operator (LASSO) operator using differential evolution (DE) algorithm. To verify the robustness of the method, numerical results for time–space fractional diffusion equation, wave equation, and Burgers' equation at different noise levels in the data are presented. Finally, the methodology is applied to a realistic example where underlying fractional differential equation associated with published experimental data obtained from an in vitro cell culture assay is learned. [DOI: 10.1115/1.4066330]

Keywords: differential evolution, fractional differential equations, sparse reconstruction

1 Introduction

Fractional differential equations (FDEs) involve fractional derivatives. In contrast to traditional differential equations that model the rate of change of a system based on integer-order derivatives, FDEs generalize this concept. Fractional-order derivatives can provide more accurate descriptions of physical phenomena that exhibit nonlocal and non-Markovian behavior [1,2]. For example, many natural phenomena, such as the diffusion of particles in porous media [3], exhibit subdiffusive or super-diffusive behavior that cannot be accurately captured by traditional integer-order differential equations. FDEs can help to provide more accurate descriptions of these phenomena, which can be useful in a range of scientific and engineering applications. Additionally, FDEs have been used in the modeling of complex systems such as visco-elastic materials [4], non-Newtonian fluids, and fractal structures, among others [5,6]. Multiterm fractional differential equations involve more than one fractional derivative. These equations are commonly used to model complex systems that exhibit multiple time scales and memory effects, such as in mechanics [7]. Basset equation [8] and Bagley Torvik equation [7] are examples of multiterm fractional differential equations. Discovering ("learning") these equations requires a deep understanding of the theory, experimental data and underlying phenomena associated with the dynamics being modeled.

Data-driven discovery of differential equations is a groundbreaking approach that leverages computational techniques and large datasets to uncover hidden mathematical relationships governing complex systems. In the field of learning differential equations, two main approaches are commonly employed: learning with known form and learning with unknown form. When dealing with known forms of differential equations, Brunton et al. [9] explored nonlinear systems through sparse identification. However, when implementing this approach in higher dimensions, computational limitations arise due to increased costs. To address the challenges of learning with known form of differential equations, researchers have turned to neural networks called physics-informed neural networks. These networks have been utilized to learn the parameters of differential equations with known forms. However, training these networks incurs a significant increase in computational costs. More recently, the work of Ren et al. [10] has emerged, focusing on learning differential equations with unknown form. This approach allows the discovery of the underlying form of differential equations directly from the data, providing a more flexible and exploratory framework for modeling complex systems.

Motivated by the wide-ranging applications of FDEs in diverse areas, researchers have begun to investigate methods for learning these types of equations too. Data-driven discovery of FDEs poses a

[1]Corresponding author.

Manuscript received April 24, 2024; final manuscript received August 20, 2024; published online September 21, 2024. Assoc. Editor: Chandramouli Padmanabhan.

challenge due to the complexity of the integral operators defining fractional order derivatives. Therefore, selecting an appropriate machine learning technique is crucial. In the learning-with-unknown-form approach, the objective is to solve differential equations when the specific form or structure is not known beforehand. This approach emphasizes exploration to find solutions or approximations without relying on predetermined mathematical expressions. Schaeffer [11] employed the Sparse reconstruction technique, utilizing Taylor expansion, to learn the partial differential equations (PDEs) using numerical optimization. Specifically, they utilized the Douglas-Rachford algorithm in their process. However, applying this algorithm to time and space fractional derivatives is challenging due to the variability in the feature matrix and the resulting nonlinearity of the system of equations. To address this issue, Singh et al. [12] introduced the use of differential evolution (DE) for minimizing the least absolute shrinkage and selection operator (LASSO) operator in order to learn time fractional partial differential equations (fPDE). Their method effectively learned the temporal Caputo fractional order and the underlying fPDE from given data, even in the presence of different levels of noise. This marked the first application of differential evolution for learning time fractional partial differential equations.

Differential evolution has gained widespread usage in solving optimization problems across various domains, including image processing [13], energy conservation [14], text classification [15], and electromagnetics [16]. Building upon the work of Singh et al. [12], we extend the application of differential evolution to the realm of time–space fractional differential equations where the aim is to learn the specific structure of the FDE. In related research on learning known-form-FDEs, Gulian et al. [17] employed Gaussian process regression to learn space fractional differential equations, Pang et al. [18] utilized physics-informed neural networks to learn time–space fractional differential equations, with a specific focus on fractional advection–diffusion equations and Rudy et al. [19] addressed learning equations using finite difference and polynomial approximation; however, this approach yielded poor performance when confronted with noisy data; in addition, their use of neural networks was restricted to specific types of fPDEs. Notably, there is currently no published result on learning time–space fractional differential equations directly from given data when the form of the FDE is not known. This highlights the gap in existing research and motivates the investigation into learning of time–space fractional differential equations using differential evolution. By leveraging the power of this optimization algorithm, we aim to overcome the limitations and address the challenges associated with learning these complex equations from data.

In this study, the research conducted by Singh et al. [12] is extended to address the problem of time–space fractional diffusion equation, wave equation, fractional Burgers' equation and an underlying fractional differential equation associated with measured data. One unique aspect of the approach is dynamic behavior of the feature matrix, which undergoes changes after each iteration. Differential evolution together with sparse reconstruction based approach proved to be particularly effective in handling the updates of the feature matrix.

This paper is structured as follows: Sec. 2 provides an overview of the sparse reconstruction technique applied to time–space fractional differential equations, utilizing the Taylor series approach and a feature matrix in every iteration. In Sec. 3, the application of differential evolution is discussed, along with modifications made to the algorithm. Section 4 presents the numerical experiments and simulation results, showcasing the effectiveness of the proposed approach through various examples. Section 5 offers concluding remarks and outline future directions for research.

2 Methods

2.1 Sparse Reconstruction of Time-Space Fractional Differential Equations. The definition of the Caputo derivative of order $m-1<\alpha<m$ with respect to t where m is a positive integer is

$$ {}_C D_{0,t}^{\alpha}\mathcal{V}(x,t)=\frac{1}{\Gamma(m-\alpha)}\int_0^t \frac{\mathcal{V}^{(m)}(x,s)}{(t-s)^{1+\alpha-m}}ds $$

We assume that the given function $\mathcal{V}(x,t)$ satisfies an equation of the following general structure

$$ {}_C D_{0,t}^{\alpha}\mathcal{V}(x,t)=\mathcal{H}(\mathcal{V},\mathcal{V}_{0,x}^{\alpha_1},\mathcal{V}_{0,x}^{\alpha_2},\ldots\mathcal{V}_{0,x}^{\alpha_n})+\mathcal{R}(x,t) \tag{2.1} $$

where $\mathcal{H}$ is the form of time–space fractional differential equation. Schaeffer [11] used Sparse Optimization to learn the partial differential equations from a given solution. This technique is extended to learn the form $\mathcal{H}$ for the given function $\mathcal{V}=\mathcal{V}(x,t)\in\mathbb{R}$ where $x\in\mathbb{R}$ and $t\geq 0$. Here, the function $\mathcal{R}=\mathcal{R}(x,t)$ represents the source term and the notation $\mathcal{V}_{0,x}^{\alpha_i}$ is used for the spatial derivatives of order $\alpha_1,\ldots\alpha_n$ such that $m_i-1<\alpha_i<m_i$ for $m_i\in\mathbb{N}$. The spatial fractional derivatives can be of various types such as Riemann–Liouville, Caputo, Riesz as explicitly mentioned in the considered examples. For simplicity, the class of fractional differential equations with two fractional orders β and γ in space ($\beta=\alpha_1$ and $\gamma=\alpha_2$) is taken into consideration which is given by

$$ {}_C D_{0,t}^{\alpha}\mathcal{V}(x,t)-\mathcal{R}(x,t)=\mathcal{H}(\mathcal{V},\mathcal{V}_{0,x}^{\beta},\mathcal{V}_{0,x}^{\gamma}) \tag{2.2} $$

The goal is to learn the form of $\mathcal{H}$, the time-fractional order α, and the space-fractional orders β and γ from the given solution $\mathcal{V}$. Note that β and γ are not equal unless explicitly stated in the considered examples. The function $\mathcal{H}$ is approximated by its second-order Taylor series expansion about the origin $\mathbf{0}=(0,0,0)$

$$ \begin{aligned} \mathcal{H}\left(\mathcal{V},\mathcal{V}_{0,x}^{\beta},\mathcal{V}_{0,x}^{\gamma}\right)\approx{}&\mathcal{H}(\mathbf{0})+\mathcal{V}\frac{\partial\mathcal{H}}{\partial\mathcal{V}}(\mathbf{0})+\mathcal{V}_{0,x}^{\beta}\frac{\partial\mathcal{H}}{\partial\mathcal{V}_{0,x}^{\beta}}(\mathbf{0})+\mathcal{V}_{0,x}^{\gamma}\frac{\partial\mathcal{H}}{\partial\mathcal{V}_{0,x}^{\gamma}}(\mathbf{0})+\frac{1}{2}\left[\mathcal{V}^2\frac{\partial^2\mathcal{H}}{\partial\mathcal{V}^2}(\mathbf{0})+\left(\mathcal{V}_{0,x}^{\beta}\right)^2\frac{\partial^2\mathcal{H}}{\left(\partial\mathcal{V}_{0,x}^{\beta}\right)^2}(\mathbf{0})+\right.\\ &\left.+\left(\mathcal{V}_{0,x}^{\gamma}\right)^2\frac{\partial^2\mathcal{H}}{\left(\partial\mathcal{V}_{0,x}^{\gamma}\right)^2}(\mathbf{0})+2\mathcal{V}\mathcal{V}_{0,x}^{\beta}\frac{\partial^2\mathcal{H}}{\partial\mathcal{V}\partial\mathcal{V}_{0,x}^{\beta}}(\mathbf{0})+2\mathcal{V}\mathcal{V}_{0,x}^{\gamma}\frac{\partial^2\mathcal{H}}{\partial\mathcal{V}\partial\mathcal{V}_{0,x}^{\gamma}}(\mathbf{0})+2\mathcal{V}_{0,x}^{\beta}\mathcal{V}_{0,x}^{\gamma}\frac{\partial^2\mathcal{H}}{\partial\mathcal{V}_{0,x}^{\beta}\partial\mathcal{V}_{0,x}^{\gamma}}(\mathbf{0})\right] \end{aligned} \tag{2.3} $$

Using approximation (2.3) in (2.2), we obtain

$$ \begin{aligned} {}_C D_{0,t}^{\alpha}\mathcal{V}(x,t)-\mathcal{R}(x,t)&\approx a_1+\mathcal{V}a_2+\mathcal{V}_{0,x}^{\beta}a_3+\cdots+\mathcal{V}_{0,x}^{\beta}\mathcal{V}_{0,x}^{\gamma}a_{10}=,\\ &=\left[1\quad\mathcal{V}\quad\mathcal{V}_{0,x}^{\beta}\quad\mathcal{V}_{0,x}^{\gamma}\quad\mathcal{V}^2\quad(\mathcal{V}_{0,x}^{\beta})^2\quad(\mathcal{V}_{0,x}^{\gamma})^2\quad\mathcal{V}\mathcal{V}_{0,x}^{\beta}\quad\mathcal{V}\mathcal{V}_{0,x}^{\gamma}\quad\mathcal{V}_{0,x}^{\beta}\mathcal{V}_{0,x}^{\gamma}\right]^{\mathrm{T}}\cdot\mathbf{a} \end{aligned} \tag{2.4} $$

where the following notation for the partial derivatives of $\mathcal{H}$ is used: $a_1 = \mathcal{H}(\mathbf{0}), a_2 = \frac{\partial \mathcal{H}}{\partial \mathcal{V}}(\mathbf{0}), \ldots, a_{10} = \frac{\partial^2 \mathcal{H}}{\partial \mathcal{V}_{0,x}^{\beta} \partial \mathcal{V}_{0,x}^{\gamma}}(\mathbf{0})$ and the right-hand side is written in vector notation as $\mathbf{a} = [a_1, a_2, a_3, a_4, a_5, a_6, a_7, a_8, a_9, a_{10}]$.

For the numerical tests, the data are generated from the known exact solution of a time–space fractional differential equation, namely, its values and fractional derivatives at the points (x_i, t_j), $i = 1\ldots m$, $j = 1\ldots n$. These data are used to assemble the feature matrix for every t_j given by

$$\mathbf{P}_j^{\beta,\gamma} := \begin{pmatrix} 1 & \mathcal{V}(x_1,t_j) & \mathcal{V}_{0,x}^{\beta}(x_1,t_j) & \mathcal{V}_{0,x}^{\gamma}(x_1,t_j) & \mathcal{V}^2(x_1,t_j) & (\mathcal{V}_{0,x}^{\beta})^2(x_1,t_j) & (\mathcal{V}_{0,x}^{\gamma})^2(x_1,t_j) & \mathcal{V}\mathcal{V}_{0,x}^{\beta}(x_1,t_j) & \mathcal{V}\mathcal{V}_{0,x}^{\gamma}(x_1,t_j) & \mathcal{V}_{0,x}^{\beta}\mathcal{V}_{0,x}^{\gamma}(x_1,t_j) \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & \mathcal{V}(x_i,t_j) & \mathcal{V}_{0,x}^{\beta}(x_i,t_j) & \mathcal{V}_{0,x}^{\gamma}(x_i,t_j) & \mathcal{V}^2(x_i,t_j) & (\mathcal{V}_{0,x}^{\beta})^2(x_i,t_j) & (\mathcal{V}_{0,x}^{\gamma})^2(x_i,t_j) & \mathcal{V}\mathcal{V}_{0,x}^{\beta}(x_i,t_j) & \mathcal{V}\mathcal{V}_{0,x}^{\gamma}(x_i,t_j) & \mathcal{V}_{0,x}^{\beta}\mathcal{V}_{0,x}^{\gamma}(x_i,t_j) \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & \mathcal{V}(x_n,t_j) & \mathcal{V}_{0,x}^{\beta}(x_n,t_j) & \mathcal{V}_{0,x}^{\gamma}(x_n,t_j) & \mathcal{V}^2(x_n,t_j) & (\mathcal{V}_{0,x}^{\beta})^2(x_n,t_j) & (\mathcal{V}_{0,x}^{\gamma})^2(x_n,t_j) & \mathcal{V}\mathcal{V}_{0,x}^{\beta}(x_n,t_j) & \mathcal{V}\mathcal{V}_{0,x}^{\gamma}(x_n,t_j) & \mathcal{V}_{0,x}^{\beta}\mathcal{V}_{0,x}^{\gamma}(x_n,t_j) \end{pmatrix}$$

The matrix $\mathbf{P}_j^{\beta,\gamma}$ consists of feature vectors as its columns. The number of columns in the matrix is the number of terms in the Taylor series expansion. In this case, there are 10 columns as there are two space fractional orders. The matrix $\mathbf{P}_j^{\beta,\gamma}$ can be reduced or extended taking into consideration a smaller or larger number of fractional derivatives. Ultimately, we seek to determine $\mathbf{a}$, α, β, and γ such that

$$\begin{aligned} &{}_C D_{0,t}^{\alpha} \mathcal{V}(x_i,t_j) - \mathcal{R}(x_i,t_j) \\ &\approx a_1 + \mathcal{V}(x_i,t_j)a_2 + \mathcal{V}_{0,x}^{\beta}(x_i,t_j)a_3 + \mathcal{V}_{0,x}^{\gamma}(x_i,t_j)a_4 \\ &\quad + \mathcal{V}^2(x_i,t_j)a_5 + (\mathcal{V}_{0,x}^{\beta})^2(x_i,t_j)a_6 \\ &\quad + (\mathcal{V}_{0,x}^{\gamma})^2(x_i,t_j)a_7 + \mathcal{V}\mathcal{V}_{0,x}^{\beta}(x_i,t_j)a_8 + \mathcal{V}\mathcal{V}_{0,x}^{\gamma}(x_i,t_j)a_9 \\ &\quad + \mathcal{V}_{0,x}^{\beta}\mathcal{V}_{0,x}^{\gamma}(x_i,t_j)a_{10} \end{aligned} \tag{2.5}$$

for all data points with indices $1 \le i \le m$ and $1 \le j \le n$. For a particular point in time with index j, the vector $\mathbf{M}_j(\alpha)$ is defined by

$$\mathbf{M}_j(\alpha) = \begin{pmatrix} {}_C D_{0,t}^{\alpha}\mathcal{V}(x_1,t_j) - \mathcal{R}(x_1,t_j) \\ \vdots \\ {}_C D_{0,t}^{\alpha}\mathcal{V}(x_m,t_j) - \mathcal{R}(x_m,t_j) \end{pmatrix}$$

The function to be minimized with respect to the unknown parameters α, β, γ, and $\mathbf{a}$ is given by

$$\mathcal{J}(\alpha,\beta,\gamma,\mathbf{a}) = \frac{1}{2}\sum_{j=1}^{n} ||\mathbf{M}_j(\alpha) - \mathbf{P}_j^{\beta,\gamma}\mathbf{a}||_2^2 + \lambda\left(||\mathbf{a}||_1 + \alpha + \beta + \gamma\right) \tag{2.6}$$

where $\lambda > 0$ is the regularization parameter in the l^1 regularization. Note that l^1 regularization promotes sparsity by imposing a constant preference (the derivative of the regularization term with respect to the parameter) for small parameter values. In that sense, feature selection is built into the l^1 regularization technique. In l^2 regularization, on the other hand, that preference for smaller parameter values itself becomes small as parameters approach zero. l^2 regularization therefore imposes parameter values close to 0 but not exactly 0.

When dealing with time-fractional differential equations, the feature matrix depends only on t but here it also depends on the space fractional orders which adds some complexity. Differential evolution deals with that efficiently by updating the feature matrix after each iteration by generating space fractional orders.

2.2 Differential Evolution. To find minimizers of (2.6), DE is used. DE is an iterative optimization algorithm developed by Storn and Price [20] designed to solve nonlinear optimization problems. It has a unique advantage over other optimization methods, such as Gradient Descent, as it does not require the objective function to be differentiable which vastly facilitates the numerical treatment of (2.6). This algorithm is used to find minimizers $(\alpha, \beta, \gamma, \mathbf{a})$ of the optimization problem (2.6) as follows:

Initially, the population matrix $\mathbf{\Omega}^0 \in \mathbf{R}^{N\times M}$ ($\mathbf{R}^{N\times M}$ denotes the space consisting of matrices of dimension $N \times M$ with each element from $\mathbf{R}$) sampling $N = 50$ sets of parameters $\alpha_i, \beta_i, \gamma_i, \mathbf{a}_i$ $(1 \le i \le N)$ is assembled. If $\beta = \gamma$, then $M = 12$ otherwise $M = 13$. Each parameter is sampled from a uniform distribution in a specific interval, e.g., (α_1, α_2), (β_1, β_2). This differs from traditional DE, where typically the same admissible interval is used for all parameters. Let $\omega_i^0 \in \mathbf{R}^M$ ($\mathbf{R}^M$ is the row vector of M elements with each element from $\mathbf{R}$) denote the parameter set $\alpha_i, \beta_i, \gamma_i, \mathbf{a}_i$, i.e., ω_i^g, $1 \le i \le N$ are the rows of the population matrix $\mathbf{\Omega}^g$ where $g = 0, 1, \ldots$ denotes the generation.

Then, the following steps are iterated until a stopping condition is reached:

(1) The standard mutation and crossover steps of DE are applied. In the mutation process, for each parameter set ω_i^g termed target vector in the population matrix, three other parameter sets of the same generation g are selected randomly. The component-wise difference of the first two, multiplied by a scalar weight, is added to the third one which yields the so-called donor vector ν_i^g. In every iteration, the weight or scaling factor is sampled from a uniform distribution in the interval $[0.2, 0.8]$. Every component of the donor vector is compared to the lower bound of its admissible interval. If a given component of ν_i^g is smaller than the lower bound, then this value of the donor vector is replaced by its lower bound. Analogously, if any component of the donor vector is larger than its upper bound, it is replaced by the upper bound.

(2) In the crossover step, a trial vector is obtained for each target vector ω_i^g in the population matrix. For this step, let $Y^{i,j}$ $(1 \le i \le N,\ 1 \le j \le M)$ be sampled from the uniform distribution on the unit interval, J_r be an integer value sampled from the uniform distribution of integers between 1 and M, and C_p is the recombination probability (in this study we use $C_p = 0.9$). Then, the trial vector u_i^g is given by

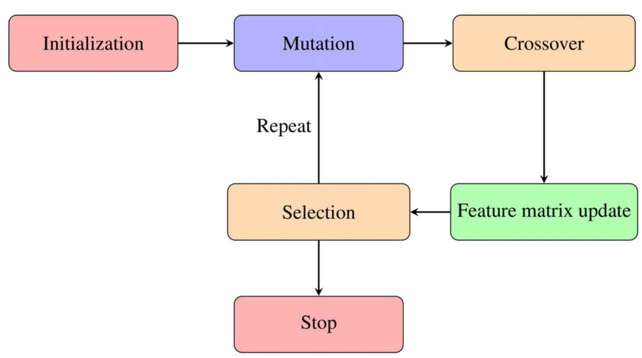


Fig. 1 Main steps involved in differential evolution (DE)

$$u_{i,j}^g = \begin{cases} v_{i,j}^g & \text{if} \quad Y^{i,j} \le C_p \quad \text{or} \quad j = J_r \\ \omega_{i,j}^g & \text{otherwise} \end{cases}$$

(3) For every target vector ω_i^g and every trial vector u_i^g, the feature matrix $\mathbf{P}_j^{\beta,\gamma}$ is calculated using the exact solution at the points (t_j, x_i). This differs from the case of time-fractional differential equations, where the feature matrix doesn't change between iterations due to the absence of space-fractional derivatives.

(4) Using the LASSO operator (2.6), the error values $\mathcal{J}(\omega_i^g)$ and $\mathcal{J}(u_i^g)$ for all target vectors and all trial vectors are calculated. Based on these error values, the parameter sets ω_i^{g+1} are as follows:

$$\omega_i^{g+1} = \begin{cases} \omega_i^g & \text{if} \quad \mathcal{J}(\omega_i^g) < \mathcal{J}(u_i^g) \\ u_i^g & \text{if} \quad \mathcal{J}(\omega_i^g) \ge \mathcal{J}(u_i^g) \end{cases}$$

(5) The next iteration is started by going back to 1 unless a threshold condition is reached. In the case of non-noisy data, iterations are stopped as soon as at least for one parameter set the error value $\mathcal{J}$ is below a given threshold value; otherwise, with noisy data, the procedure after 1000 iterations is stopped.

From the last generation g, the parameter set ω_i^g with lowest approximation error $\mathcal{J}(\omega_i^g)$ is selected. It contains the fractional orders and coefficients determining the shape $\mathcal{H}$ of the learned equation. By following these steps, the DE algorithm is utilized to effectively learn the desired fractional orders and coefficients, and to recover the original form of the equation. Figure 1 summarizes the above procedure.

3 Results

To illustrate the effectiveness of the proposed method (discussed in Secs. 2.1 and 2.2), numerical examples are presented. For the first example, an equation with spatial derivatives of Riemann–Liouville (RL) type is considered, while in the second example, an equation involving Riesz fractional derivative in space is considered. By considering different types of spatial fractional derivatives in these two examples, the robustness of the proposed method across various noise levels is assessed.

The third example is more intricate since it includes nonlinear terms in the equation. This is used to explore a larger number of spatial fractional orders, as compared to the preceding examples. In the last example, the data describing the Fisher-KPP equation are considered and the underlying differential equation is learned using the approach described in Secs. 2.1 and 2.2. Note that in first three numerical test cases, the regularization parameter is $\lambda = 100$ while in the last two examples, regularization parameter $\lambda = 10^{-7}$ is chosen. All simulations are conducted using MATLAB on a 12th Gen Intel(R) Core(TM), i5-1235 U, 1.30 GHz processor, with 16 GB RAM. These computational resources are chosen to ensure efficient and reliable execution of the simulations.

3.1 Time–Space Fractional Diffusion Equation.

The time–space fractional diffusion equation is a generalized form of the classical diffusion equation that incorporates fractional derivatives in both time and space. It is used to model various physical and biological processes with anomalous diffusion behavior, where the diffusion process is not well described by classical diffusion equations [21,22]. The general form of the nonhomogeneous time–space fractional diffusion equation with a source term can be written as

$$\frac{\partial^\alpha u(x,t)}{\partial t^\alpha} = \nu \frac{\partial^\beta u(x,t)}{\partial x^\beta} + g(x,t)$$

where $t \ge 0$, $x \in \Omega \subset \mathbf{R}$, $u(x, t)$ is the concentration of the quantity undergoing diffusion, ν is the diffusion coefficient, α and β are the fractional orders of the time and space derivatives such that $0 < \alpha < 1$ and $1 < \beta < 2$, respectively, and $g(x, t)$ is the given, known source term that represents any additional external influence or forcing on the system. Consider the following time–space fractional boundary value problem of diffusion type for $t > 0$ on the domain $0 < x < 1$:

$$\begin{cases} {}_C D_{0,t}^\alpha u(x,t) = {}_{RL} D_{0,x}^\beta u(x,t) + g(x,t) \\ u(0,t) = u(1,t) = 0 \\ u(x,0) = 0 \end{cases} \tag{3.1}$$

where $g(x,t) = t^{9.5}\frac{\Gamma(11)}{\Gamma(10.5)}x(1-x) - t^{10}\frac{\Gamma(2)}{\Gamma(0.5)}x^{-0.5} + t^{10}\frac{\Gamma(3)}{\Gamma(1.5)}x^{0.5}$. Note that the function $\mathcal{V}(x,t) = t^{10}x(1-x)$ is the exact solution of the boundary value problem (3.1) when $\alpha = 0.5$ and $\beta = 1.5$ the definition of the RL fractional derivative is

$${}_{RL}D_{0,x}^\beta \mathcal{V}(x,t) = \frac{1}{\Gamma(2-\beta)}\frac{\partial^2}{\partial x^2}\int_0^t \frac{\mathcal{V}(x,s)}{(x-s)^{\beta-1}}ds$$

According to the approach sketched in Sec. 2.1, the values of α, β, and γ where $0 < \alpha < 1$, $1 < \beta < 2$ and $1 < \gamma < 2$ are to be learned from the exact solution $\mathcal{V}(x,t)$ of (3.1). Note that in this example, $\beta = \gamma$. The approximated value of the left side operator in Eq. (3.1) with the help of the exact solution $\mathcal{V}(x,t)$ is given by

$${}_C D_{0,t}^\alpha \mathcal{V}(x,t) - \mathcal{R}(x,t) \approx a_1 + \mathcal{V}a_2 + \mathcal{V}_{0,x}^\beta a_3 + \cdots + \mathcal{V}_{0,x}^\beta \mathcal{V}_{0,x}^\gamma a_{10} \tag{3.2}$$

where $\mathcal{R}(x,t) = g(x,t)$ and the coefficients a_i are calculated as in Eq. (2.4). The quantities $\mathcal{V}_{0,x}^\beta(x,t) = t^{10}\frac{\Gamma(2)}{\Gamma(2-\beta)}x^{1-\beta} - t^{10}\frac{\Gamma(3)}{\Gamma(3-\beta)}x^{2-\beta}$ and $\mathcal{V}_{0,x}^\gamma(x,t) = t^{10}\frac{\Gamma(2)}{\Gamma(2-\gamma)}x^{1-\gamma} - t^{10}\frac{\Gamma(3)}{\Gamma(3-\gamma)}x^{2-\gamma}$ in Eq. (3.2) are RL fractional derivatives where $\mathcal{V}(x,t)$ is the exact solution of Eq. (3.1). In the numerical discretization on the uniform grid t_j, x_i, the values $\mathcal{R}(x_i, t_j) = g(x_i, t_j) + \xi_i$ where ξ_i is normally distributed with mean 0 and variance σ^2 are calculated. It represents random noise in observing discrete values of the solution $\mathcal{V}$ on the grid (x_i, t_j) and in numerically evaluating fractional derivatives to setup the feature matrix $\mathbf{P}_j^{\beta,\gamma}$. It is evident from Sec. 2.1 that the feature matrix $\mathbf{P}_j^{\beta,\gamma}$ changes in every iteration. To test the method, the exact solution is used to generate data points on a uniform grid of 200×200 points. By applying the method to the generated data at different noise levels, the algorithm succeeds in determining the values of the time and space fractional orders, as well as the form of the diffusion equation represented by the coefficients $\mathbf{a}$. Table 1 provides a summary of the learned values at different noise levels. At all noise levels, the learning algorithm correctly identifies the form of (3.1), namely, the order of the time derivative being 0.5 and the spatial derivative being of order 1.5.

At all noise levels, the learning algorithm learns correctly one spatial order of derivatives as indicated by identical values of β and γ in Table 1. The sum of the specific coefficients $a_3 = \mathcal{V}_{0,x}^\beta(\mathbf{0})$ and $a_4 = \mathcal{V}_{0,x}^\gamma(\mathbf{0})$ at all noise levels is close to 1, which is the coefficient of the spatial derivative in (3.1), yet they differ between noise levels. This is due to the absence of strict convexity in the l^1 regularization term, which potentially admits nonunique minimizers of (2.6). The learned equation in case of noise level 0.05 is given by

$${}_C D_{0,t}^{0.45720} u(x,t) = 0.94952 \, {}_{RL} D_{0,x}^{1.52330} u(x,t) + g(x,t)$$

The obtained results illustrate the effectiveness and robustness of the method for learning the time–space fractional diffusion equation, even in the presence of noise where noise is incorporated on the left side of (3.2).

3.2 Time–Space Fractional Wave Equation.

The general form of the nonhomogeneous time–space fractional wave equation [23] for the unknown function $u(x, t)$ is given by

Table 1 Learned parameters of time–space fractional diffusion equation

σ (noise)	α	β	γ	$\mathbf{a}$ (rounded to five decimal places)
0	0.49018	1.50170	1.50170	[0.00206, 0, 0.46910, 0.50508, 0.01842, 0.00179, 0.00737, 0.00156, 0, 0]
0.05	0.45720	1.52330	1.52330	[0, 0, 0.19792, 0.75160, 0, 0, 0, 0, 0, 0]
0.10	0.45699	1.52350	1.52350	[0, 0, 0.13174, 0.93610, 0, 0, 0, 0, 0, 0]

Table 2 Learned parameters of time–space fractional wave equation

Noise	α	β	γ	$\mathbf{a}$ (rounded to five decimal places)
0	1.45780	1.50220	1.50220	[0, 0.00071, 0.00422, 0.98068, 0.00592, 0, 0.00068, 0.024813, 0.00124, 0.00007]
0.05	1.40290	1.50620	1.50620	[0, 0, 0.03928, 0.94716, 0, 0, 0, 0.01991, 0.01775, 0]
0.10	1.40440	1.50620	1.50620	[0, 0, 0.93227, 0.05431, 0, 0, 0, 0.01962, 0.017312, 0]

$$\frac{\partial^\alpha u(x,t)}{\partial t^\alpha} = c^2 \frac{\partial^\beta u(x,t)}{\partial x^\beta} + g(x,t) \tag{3.3}$$

where $t \geq 0$ represents time and $x \in \Omega \subset \mathbf{R}$ represents space coordinates. The constant c represents the wave speed. Here, $1 < \alpha < 2$ and $1 < \beta < 2$. The right-hand side of the equation includes the term $g(x, t)$, which represents a nonhomogeneous forcing function. The specific form of the forcing function depends on the particular problem or system under consideration. Some applications lie in symmetry breaking [24] and in anomalous transport [25].

The following initial-boundary value problem on the domain $0 < x < 1$ is

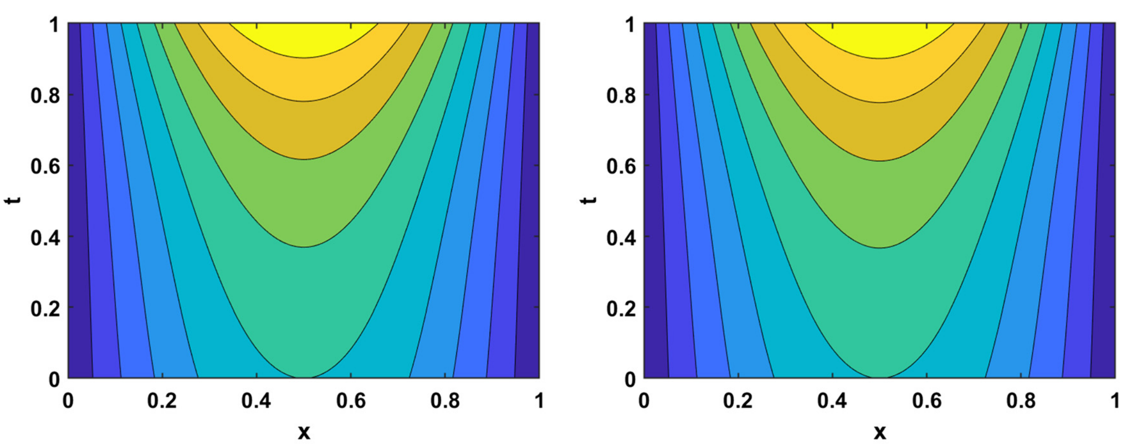
$$\begin{cases} {}_C D^\alpha_{0,t} u(x,t) = {}_{RZ}D^\beta_{0,x} u(x,t) + g(x,t) \\ u(0,t) = u(1,t) = 0 \\ u(x,0) = 2x(1-x) \\ u_t(x,0) = x(1-x) \end{cases} \tag{3.4}$$

Assuming that the inhomogeneity is given by $g(x,t) = x(1-x)\frac{\Gamma(4.5)}{\Gamma(3)}t^2 + (t^{3.5}+t+2)\frac{1}{2\cos(\beta\pi/2)} \left(\frac{\Gamma(2)}{\Gamma(0.5)}x^{-0.5} - \frac{\Gamma(3)}{\Gamma(1.5)}x^{0.5}\right) + (t^{3.5} +t+2)\frac{1}{2\cos(\beta\pi/2)}\left(\frac{\Gamma(2)}{\Gamma(0.5)}(1-x)^{-0.5} - \frac{\Gamma(3)}{\Gamma(1.5)}(1-x)^{0.5}\right)$, the exact solution to this problem is given by $\mathcal{V}(x,t) = (t^{3.5}+t+2)x(1-x)$ where $\alpha = \beta = 1.5$. The spatial derivative is a Riesz derivative given by

$${}_{RZ}D^\beta_{0,x}\mathcal{V}(x,t) = \frac{-1}{2\cos(\beta\pi/2)}\left({}_{RL}D^\beta_{0,x}\mathcal{V}(x,t) + {}_{RL}D^\beta_{x,1}\mathcal{V}(x,t)\right)$$

Note that here $\beta = \gamma$. Similar to the approach in Sec. 4.1, the aim is to learn the structure of the PDE within the class of Eq. (2.4) where $a_3 = \mathcal{V}^\beta_{0,x}(\mathbf{0})$ and $a_4 = \mathcal{V}^\gamma_{0,x}(\mathbf{0})$ in the feature matrix $\mathbf{P}^{\beta,\gamma}_j$ are interpreted as Riesz derivatives. Precise Riesz-type fractional derivatives of order $1 < \beta < 2$ in Eq. (2.4) of the exact solution $\mathcal{V}$ of Eq. (3.4) are given by $\mathcal{V}^\beta_{0,x}(x,t) = -(t^{3.5}+t+2)\frac{1}{2\cos(\beta\pi/2)}(\frac{\Gamma(2)}{\Gamma(0.5)}x^{-0.5} - \frac{\Gamma(3)}{\Gamma(1.5)}x^{0.5}) - (t^{3.5}+t+2)\frac{1}{2\cos(\beta\pi/2)}(\frac{\Gamma(2)}{\Gamma(0.5)}\ (1-x)^{-0.5} - \frac{\Gamma(3)}{\Gamma(1.5)}(1-x)^{0.5})$ which are used to generate the feature matrix in every iteration of the differential evolution algorithm.

The learning algorithm successfully identified the time and space fractional orders, as well as the specific form of the wave equation represented by the vector $\mathbf{a}$ at various noise levels. The learned value of α is close to 1.5 and the values of both, β and γ are also close to the exact value 1.5, the sum of their weights, $a_3 + a_4$ being close to 1 at all noise levels. The individual values a_3 and a_4, however, differ due to the lack of uniqueness of minimizers. This indicates that the learning algorithm correctly identifies the presence of a single spatial derivative of order 1.5. The results, summarized in Table 2, illustrate the robustness of the approach in handling noise, which is incorporated on the left side of Eq. (2.2) in the same way as above in section. The learned equation in case of noise level 0.05 (ignoring the coefficients of the terms not involved in wave equation) is given as



(a) (b)

Fig. 2 Contour plot of exact solution (a) and learned solution (b) of time–space fractional wave equation at noise level 0.05

Table 3 Learned parameters of time–space fractional Burgers' equation

Noise	α	β	γ	**a** (rounded to five decimal places)
0	0.49985	0.51122	1.48790	[0, 0, 0, 0.10043, 0.036685, 0.00002, 0, 0.96799, 0, 0]
0.05	0.45320	0.52300	1.46299	[0,0,0,0.10089,0.03764,0,0,0.95299,0,0]
0.10	0.45098	0.52486	1.46010	[0,0,0,0.10120,0.03781,0,0,0.95105,0,0]

$$ {}_{C}D_{0,t}^{1.40290}u(x,t) = 0.98644_{RZ}D_{0,x}^{1.50620}u(x,t) + g(x,t) \tag{3.5} $$

Fig. 2 shows the contour plot of the exact solution and solution of learned time space fractional wave Eq. (3.5). This numerical example demonstrates that the algorithm is effective at learning the symmetric Riesz fractional derivative in addition to nonsymmetric fractional derivatives of Riemann–Liouville and Caputo type.

3.3 Time-Space Fractional Burgers' Equation.

The general form of the time–space fractional viscid Burgers' equation [26] can be expressed as follows:

$$ \frac{\partial^\alpha u}{\partial t^\alpha} = \lambda u \frac{\partial^\beta u}{\partial x^\beta} + \nu \frac{\partial^\gamma u}{\partial x^\gamma} + g(x,t) \tag{3.6} $$

Here, the equation incorporates fractional derivatives with respect to both time t and space x where $t \geq 0$, $x \in \Omega \subset \mathbf{R}$, $0 < \alpha < 1$, $0 < \beta < 1$, and $1 < \gamma < 2$. The parameters ν and λ represent the coefficients for diffusion and flux, respectively. The function $g(x,t)$ denotes a source or forcing term. The time–space fractional Burgers' equation finds applications in various fields where the dynamics of fractional diffusion processes is involved.

Solutions of these equations have characteristics that are very different from solutions to diffusion and wave equations considered about due to the nonlinear term present in the flux. As a numerical test case, we consider the following initial-boundary value problem on the domain $0 < x < 1$

$$ \begin{cases} {}_{C}D_{0,t}^{\alpha}u(x,t) = u(x,t)_{RL}D_{0,x}^{\beta}u(x,t) + \dfrac{1}{10}{}_{RL}D_{0,x}^{\gamma}u(x,t) + g(x,t) \\ u(0,t) = 0 \\ u(1,t) = 10\pi t \\ u(x,0) = 0 \end{cases} \tag{3.7} $$

where $g(x,t) = 10\pi x^2 \frac{\Gamma(2)}{\Gamma(1.5)} t^{0.5} - \pi t \frac{\Gamma(3)}{\Gamma(1.5)} x^{0.5} - 100\pi^2 t^2 x^{3.5} \frac{\Gamma(3)}{\Gamma(2.5)}$. An exact solution of (3.7) is $\mathcal{V}(x,t) = 10\pi x^2 t$ when $\alpha = \beta = 0.5$, $\gamma = 1.5$.

As in the examples above, to find a minimizer of (2.6), feature matrix is generated using this exact solution, with a total of 200 points in both the x and t axes, using a uniform grid size in every iteration of the differential evolution algorithm. Using the learning algorithm, the values of the time and space fractional orders, α, β, and γ as well as the form of the Burgers' equation represented by $a_4 \approx 0.1$ and $a_8 \approx 1$ are successfully learned with all other coefficients close to zero. We incorporate the effect of noise in the same way as considered in Secs. 3.1 and 3.2. Table 3 provides a summary of the learned values with different noise levels. The learned equation (ignoring the terms not involved in Burgers' equation) is given as

$$ \begin{aligned} {}_{C}D_{0,t}^{0.49985}u(x,t) &= 0.96799u(x,t)_{RL}D_{0,x}^{0.51122}u(x,t) \\ &\quad + 0.10043_{RL}D_{0,x}^{1.48790}u(x,t) + g(x,t) \end{aligned} $$

The obtained results strongly indicate the robustness of the method for learning even nonlinear time–space fractional equations.

3.4 Time–Space Fractional Fisher–Kolmogorov–Petrovsky–Piskunov Equation.

The Fisher–KPP equation lies in the class of nonlinear reaction-diffusion equations. Applications of the Fisher–KPP equation can be found in tumor growth modeling, ecology, cancer treatment, and tissue engineering [27–30]. The one-dimensional time–space fractional Fisher–KPP equation is given by

$$ \frac{\partial^\alpha u}{\partial t^\alpha} = D \frac{\partial^\gamma u}{\partial x^\gamma} + ru\left(1 - \frac{u}{K}\right) \tag{3.8} $$

where $0 < \alpha < 1$ and $1 < \gamma < 2$. The parameter D is the diffusion constant, r is the growth rate of the chosen species, and K is the carrying capacity. For a numerical test, the one-dimensional Fisher–KPP equation for the function $u(x,t)$ as a model for tumor growth on the domain $-50 \leq x \leq 50$, $t \geq 0$ is given by

$$ \frac{\partial u}{\partial t} = 0.02 \frac{\partial^2 u}{\partial x^2} + 0.1u(1-u) \tag{3.9} $$

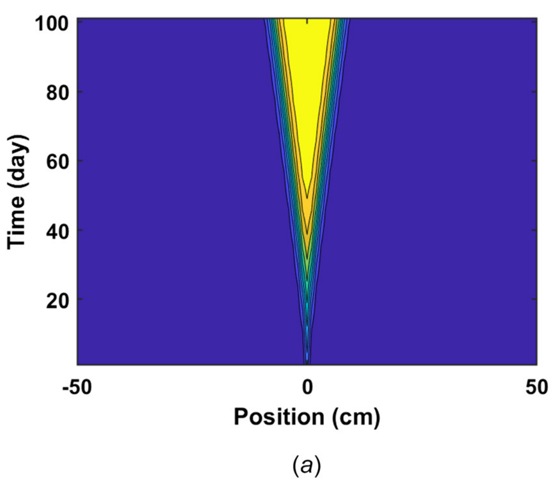


Fig. 3 Contour plot of simulated solution (*a*) and learned simulated solution (*b*) of time–space fractional Fisher–KPP equation

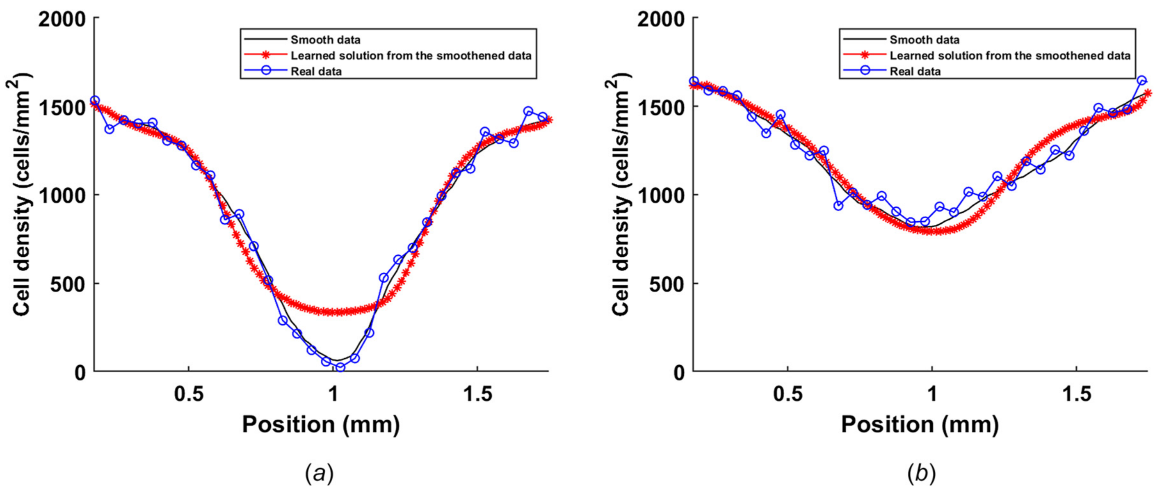


Fig. 4 **Numerical solution of learned time–space fractional Fisher–KPP equation with real and the smoothened data at (*a*) $t = 24$ h and (*b*) $t = 36$ h**

Table 4 **Learned parameters of time–space fractional Fisher-KPP equation**

Noise	α	β	γ	**a** (rounded to five decimal places)
0	0.99999	0.99999	1.99999	$[0, 0.10582, 0, 0.01708, -0.10584, 0, -0.00755, 0, -0.00198, 0]$
0.05	0.96453	0.97775	1.96988	$[0,0.09980,0,0.01698,-0.10776,0,0,0,0,0]$
0.10	0.96200	0.97639	1.96380	$[0,0.09899,0,0.01652,-0.10834,0,0,0,0,0]$

where t refers to time (in day "d") and x refers to position (in "cm"). The parameters D and r have units cm^{2}d^{-1} and d^{-1}, respectively. The initial condition for (3.9) is prescribed as

$$u(x, 0) = \begin{cases} 0.25 & -0.5 \le x \le 0.5 \\ 0 & \text{otherwise} \end{cases}$$

Eq. (3.9) is numerically simulated under periodic boundary conditions using the forward in time and central in space scheme with step-size values $dt = dx = 1$, and its solution is shown as simulated solution in Fig. 3. For testing the learning procedure, the data generated from simulated solution are used at 100×101 grid points. The Caputo fractional derivatives with respect to time and spatial variables are calculated directly from the simulated data using L1 and L2 schemes [31]. The derivatives are calculated at 100×101 grid points before substitution in the feature matrix where upper and lower bounds for elements of **a** are 1 and -1, respectively. Also, the upper and lower bounds for $\{\alpha, \beta, \gamma\}$ are $\{1, 1, 2\}$ and $\{0, 0, 1\}$, respectively. The learning algorithm successfully learned the time and space fractional orders with the particular form of Fisher–KPP equation from the simulated data (ignoring the almost zero coefficients) given as

$$\frac{\partial u}{\partial t} = 0.01708 \frac{\partial^2 u}{\partial x^2} + 0.10582u - 0.10584u^2 \tag{3.10}$$

The solution of Eq. (3.10) is "learned simulated solution." Table 4 summarizes the learned parameters of Fisher–KPP equation for different levels of noise. The identified fractional orders α and γ are very close to 1 and 2 (classical orders) although the algorithm runs over a range of values for $\alpha \in (0, 1)$ and $\beta \in (1, 2)$. The learned value of β is close to 1, indicating that the fractional derivative of order β with respect to x exists but since the coefficient of u_x is 0, this derivative is not playing any role. The values of D and r are correctly identified and noninvolved terms have coefficient 0 when rounded to 2 decimal places. Figure 3 shows that the simulated solution of (3.9) and the solution of the Fisher–KPP equation with the learned parameters (learned simulated solution) are very close. These results claim that the learning method is very accurate in determining the Fisher–KPP equation (3.9) even in the case when the exact solution is not known and a discrete solution is provided on the grid.

Table 5 **Learned parameters of time–space fractional Fisher–KPP equation**

α	γ	**a** (rounded to five decimal places)
1	1.67585	$[417.71330, 1.65177, 0, 0.02151, -0.00126, 0, 0, 0, 0, 0]$

3.5 Learning a Fractional Fisher–Kolmogorov–Petrovsky–Piskunov Equation From Data. Finally, we consider measured data representing scratch assay density profiles measured at 12 h (h) intervals as obtained by Jin et al. [32]. The data is provided at 38×5 grid points with time-step and space-step values of 12 h and 50 μm. Note that for numerical convenience, we convert the units to mm and days, respectively. The new step sizes are 0.5 d and 0.05 mm. The data at two different times is shown in Fig. 4. To facilitate taking numerical derivatives the data are smoothened using kernel smoothing and evaluated on a much finer grid with step sizes of 0.0091667d and 0.0159 mm. The smoothened data are shown in Fig. 4. In this application, we assume that only one spatial derivative may be fractional and we keep the order of the time derivative at 1. To this end, we impose $a_3 = a_6 = a_8 = a_{10} = 0$ in (2.4) to avoid the presence of fractional derivatives of order β. The value of the space fractional derivative of order between 1 and 2 is calculated by taking a classical derivative of a spatial fractional derivative of order between 0 and 1. Table 5 summarizes the learned parameters in this case. The learned equation is

$$\frac{\partial u}{\partial t} = 0.02151132 \frac{\partial^{1.6758506}}{\partial x^{1.6758506}} u + 1.651772u - 0.00126411u^2 + 417.7133 \tag{3.11}$$

Equation (3.11) is numerically simulated using the finite difference method where the spatial fractional derivative is approximated using the fractional trapezoidal method due its higher order of accuracy as compared to the L2 scheme [31]. The (classical) time derivative is approximated using the forward difference approach.

As initial and boundary values, we use the values given by the smoothened data. The resulting solution of Eq. (3.11) is what we call "learned solution from the smoothened data," which is shown in Fig. 4. It compares well with the data and we therefore conclude that the algorithm is successful in learning the equation underlying this phenomenon.

4 Discussion and Conclusion

In this study, we have proposed a novel approach for learning time–space fractional differential equations using a combination of sparse reconstruction and differential evolution. The objective of the research is to accurately capture the dynamics of time–space fractional diffusion, wave, and Burgers' equations, both in the presence and absence of noise. Through numerical experiments, the effectiveness and accuracy of the proposed approach is demonstrated. The combination of sparse reconstruction techniques helped to efficiently extract relevant features from the given data, while the application of differential evolution facilitated the optimization process for determining the parameters of the fractional differential equations. The learned models are able to accurately capture the intricate dynamics of time–space fractional systems, even in the presence of noisy input data. A real-world example is also included and the underlying differential equation is learned. The solution to this equation approximates the data well although we do not attempt to minimize the error, but the residual of the underlying equation when evaluated at the (smoothened) data. The advantage of this approach is that we never have to solve numerically the forward problem. This suggests that the proposed approach is well-suited for real-world scenarios where data may be subject to noise and uncertainty.

Looking ahead, there are several promising directions for future research. First, the approach can be extended to higher dimensions, allowing for the modeling and analysis of complex systems in multiple spatial dimensions. Second, the integration of additional constraints and regularization techniques could further enhance the accuracy and stability of the learned models. Third, the applicability of the methodology can be extended to other types of real-world data as well. Fourth, the concept of learning the structure of fractional models underlying given data should also be evaluated through classical error minimization in combination with inverse problem regularization techniques. Finally, in the presence of high noise in real world data, proper smoothing techniques need to be implemented in both dimensions to calculate time and space fractional orders of derivatives, which is a notable challenge that we wish to overcome in the near future.

In conclusion, the research demonstrates the effectiveness of the sparse reconstruction and differential evolution approach in learning time–space fractional differential equations. The ability to accurately learn and analyze time–space fractional diffusion, wave, and Burgers' equations opens up new avenues for applications in diverse fields, such as physics, engineering, and finance. The findings presented in this study contribute to the field of fractional calculus and provide a solid foundation for further advancements in the understanding and modeling of fractional systems.

Funding Data

- Department of Science and Technology, India (Grant No. SR/FST/MS-1/2019/45; Funder ID: 10.13039/501100001409).

Data Availability Statement

The authors attest that all data for this study are included in the paper.

References

[1] Teka, W., Upadhyay, R. K., and Mondal, A., 2017, "Fractional-Order Leaky Integrate-and-Fire Model With Long-Term Memory and Power Law Dynamics," Neural Networks, **93**, pp. 110–125.
[2] Kheiri, H., and Jafari, M., 2018, "Optimal Control of a Fractional-Order Model for the HIV/AIDS Epidemic," Int. J. Biomath., **11**(07), p. 1850086.
[3] Ali, I., Malik, N., and Chanane, B., 2014, "Fractional Diffusion Model for Transport Through Porous Media," 5th International Conference on Porous Media and Their Applications in Science, Kona, HI, June 22–27.
[4] Peng, Y., Zhao, J., Sepehrnoori, K., and Li, Z., 2020, "Fractional Model for Simulating the Viscoelastic Behavior of Artificial Fracture in Shale Gas," Eng. Fract. Mech., **228**, p. 106892.
[5] Carréra, Y., Avila-de la Rosa, G., Vernon-Carter, E. J., and Alvarez-Ramirez, J., 2017, "A Fractional-Order Maxwell Model for Non-Newtonian Fluids," Phys. A, **482**, pp. 276–285.
[6] Stanisauskis, E., Mashayekhi, S., Pahari, B., Mehnert, M., Steinmann, P., and Oates, W., 2022, "Fractional and Fractal Order Effects in Soft Elastomers: Strain Rate and Temperature Dependent Nonlinear Mechanics," Mech. Mater., **172**, p. 104390.
[7] Torvik, P., and Bagley, R., 1984, "On the Appearance of the Fractional Derivative in the Behavior of Real Materials," ASME J. Appl. Mech., **51**(2), pp. 294–298.
[8] Mainardi, F., 2012, "Fractional Calculus: Some Basic Problems in Continuum and Statistical Mechanics," Fractals and Fractional Calculus in Continuum Mechanics, Springer, Vienna, pp. 291–348.
[9] Brunton, S., Proctor, J., and Kutz, J., 2016, "Discovering Governing Equations From Data: Sparse Identification of Nonlinear Dynamical Systems," Proc. Natl. Acad. Sci., **113**(15), pp. 3932–3937.
[10] Ren, P., Rao, C., Liu, Y., Wang, J.-X., and Sun, H., 2022, "Phycrnet: Physics-Informed Convolutional-Recurrent Network for Solving Spatiotemporal PDEs," Comput. Methods Appl. Mech. Eng., **389**, p. 114399.
[11] Schaeffer, H., 2017, "Learning Partial Differential Equations Via Data Discovery and Sparse Optimization," Proc. R. Soc. A, **473**(2197), p. 20160446.
[12] Singh, A. K., Mehra, M., and Alikhanov, A. A., 2022, "Data-Driven Discovery of Time Fractional Differential Equations," Computational Science – ICCS 2022, London, UK, June 21–23, pp. 56–63.
[13] Chakraborty, S., Saha, A. K., Ezugwu, A. E., Agushaka, J. O., Zitar, R. A., and Abualigah, L., 2023, "Differential Evolution and Its Applications in Image Processing Problems: A Comprehensive Review," Arch. Comput. Methods Eng., **30**(2), pp. 985–1040.
[14] He, D., Zhang, L. Z., Songlin, G., Chen, Y., Shan, S., and Jian, H., 2021, "Energy-Efficient Train Trajectory Optimization Based on Improved Differential Evolution Algorithm and Multi-Particle Model," J. Cleaner Prod., **304**, p. 127163.
[15] Diab, D. M., and Hindi, K., 2017, "Using Differential Evolution for Fine Tuning Naïve Bayesian Classifiers and Its Application for Text Classification," Appl. Soft Comput., **54**, pp. 183–199.
[16] Chen, Y., Yang, S., and Nie, Z., 2007, "Synthesis of Uniform Amplitude Thinned Linear Phased Arrays Using the Differential Evolution Algorithm," Electromagnetics, **27**(5), pp. 287–297.
[17] Gulian, M., Raissi, M., Perdikaris, P., and Karniadakis, G., 2018, "Machine Learning of Space-Fractional Differential Equations," SIAM J. Sci. Comput., **41**(4).
[18] Pang, G., Lu, L., and Karniadakis, G., 2019, "fPINNs: Fractional Physics-Informed Neural Networks," SIAM J. Sci. Comput., **41**(4), pp. A2603–A2626.
[19] Rudy, S., Brunton, S., Proctor, J., and Kutz, J., 2016, "Data-Driven Discovery of Partial Differential Equations," Sci. Adv., **3**(4), p. e1602614.
[20] Storn, R., and Price, K., 1997, "Differential Evolution - A Simple and Efficient Heuristic for Global Optimization Over Continuous Spaces," J. Global Optim., **11**, pp. 341–359.
[21] Aguilar, J.-P., Korbel, J., and Luchko, Y., 2019, "Applications of the Fractional Diffusion Equation to Option Pricing and Risk Calculations," Mathematics, **7**(9), p. 796.
[22] Rida, S., El-Sayed, A., and Arafa, A., 2010, "Effect of Bacterial Memory Dependent Growth by Using Fractional Derivatives Reaction-Diffusion Chemotactic Model," J. Stat. Phys., **140**(4), pp. 797–811.
[23] Li, Y., Wang, Y., and Deng, W., 2017, "Galerkin Finite Element Approximations for Stochastic Space-Time Fractional Wave Equations," SIAM J. Numer. Anal., **55**(6), pp. 3173–3202.
[24] Ibrahim, R., and Baleanu, D., 2021, "Symmetry Breaking of a Time-2D Space Fractional Wave Equation in a Complex Domain," Axioms, **10**(3), p. 141.
[25] Liu, L., Zhang, S., Chen, S., Liu, F., Feng, L., Turner, I., Zheng, L., and Zhu, J., 2023, "An Application of the Distributed-Order Time- and Space-Fractional Diffusion-Wave Equation for Studying Anomalous Transport in Comb Structures," Fractal Fractional, **7**(3), p. 239.
[26] Saad, K., and Al-Sharif, E., 2017, "Analytical Study for Time and Time-Space Fractional Burgers' Equation," Adv. Differ. Equations, **2017**(1), pp. 1–15.
[27] Henscheid, N., 2020, "Generating Patient-Specific Virtual Tumor Populations With Reaction-Diffusion Models and Molecular Imaging Data," Math. Biosci. Eng., **17**(6), pp. 6531–6556.
[28] Paul, G., Tauhida, T., and Kumar, D., 2022, "Revisiting Fisher-KPP Model to Interpret the Spatial Spreading of Invasive Cell Population in Biology," Heliyon, **8**(10), p. e10773.
[29] Contri, B., 2018, "Fisher-KPP Equations and Applications to a Model in Medical Sciences," Networks Heterog. Media, **13**(1), pp. 119–153.
[30] Habbal, A., Barelli, H., and Malandain, G., 2014, "Assessing the Ability of the 2D Fisher-KPP Equation to Model Cell-Sheet Wound Closure," Math. Biosci., **252**, pp. 45–59.
[31] Li, C., and Zeng, F., 2015, *Numerical Methods for Fractional Calculus*, Chapman & Hall/CRC Numerical Analysis and Scientific Computing Series, CRC Press, New York.
[32] Jin, W., Shah, E., Penington, C., McCue, S., Chopin, L., and Simpson, M., 2016, "Reproducibility of Scratch Assays is Affected by the Initial Degree of Confluence: Experiments, Modelling and Model Selection," J. Theor. Biol., **390**, pp. 136–145.