Nonlinear Dyn (2023) 111:12421–12431
https://doi.org/10.1007/s11071-023-08463-1

ORIGINAL PAPER

Inferring the fractional nature of Wu Baleanu trajectories

J. Alberto Conejero · Òscar Garibo-i-Orts · Carlos Lizama

Received: 25 October 2022 / Accepted: 5 April 2023 / Published online: 9 May 2023

Abstract We infer the parameters of fractional discrete Wu Baleanu time series by using machine learning architectures based on recurrent neural networks. Our results shed light on how clearly one can determine that a given trajectory comes from a specific fractional discrete dynamical system by estimating the fractional exponent and the growth parameter μ. With this example, we also show how machine learning methods can be incorporated into the study of fractional dynamical systems.

Keywords Fractional dynamical systems · Chaotic systems · Machine learning · Recurrent neural networks

Mathematics Subject Classification 26A33 · 65P20 · 68U20

JAC acknowledges funding from grant PID2021-124618NB-C21 funded by MCIN/AEI/ 10.13039 /501100011033 and by "ERDF A way of making Europe," by the "European Union." We also thank funding for the open-access charges from CRUE-Universitat Politècnica de València.
CL was partially supported by ANID under FONDECYT Grant Number 1220036.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/s11071-023-08463-1.

J. A. Conejero (✉)
Instituto Universitario de Matemática Pura y Aplicada, Universitat Politècnica de València, Camí de Vera, s/n, 46022 València, Spain
e-mail: aconejero@upv.es

Ò. Garibo-i-Orts
GRID - Grupo de Investigación en Ciencia de Datos, Valencian International University - VIU, Carrer Pintor Sorolla 21, 46002 València, Spain
e-mail: oscar.garibo@campusviu.es

C. Lizama
Departamento de Matemática y Ciencia de la Computación, Universidad de Santiago de Chile, Las Sophoras 173, Estación Central, Santiago, Chile
e-mail: carlos.lizama@usach.cl

1 Introduction

The logistic equation introduced by May [27] models the behavior of a population that grows exponentially, but some constraints of the environment limit this growth. We can express it as

$$v(n+1) = \eta v(n)(1 - v(n)), \quad \text{for } n \in \mathbb{N}_0, \tag{1}$$

where $v(0) \in [0, 1]$ and $\eta \in \mathbb{R}$. This equation provides the simplest example of a one-parameter nonlinear dynamical system with nontrivial dynamics. It is very well-known that if $0 \leq \eta \leq 4$, we have a well-defined dynamical system on $[0, 1]$. For $\eta > 4$, we still can have a discrete dynamical system, but this will only be defined on the complementary of a particular Cantor set in $[0, 1]$; see for instance [34].

In order to incorporate some difference operator that we can later extend naturally with a discrete fractional

derivative, we can transform the logistic equation by applying the change of variable $v(n) = \frac{\eta}{\eta-1}u(n)$. In this way, instead of having a formula for computing the term $u(n+1)$ by recurrence, we express that the forward Euler operator Δ is equal to the nonlinear right term of the logistic, that is

$$\Delta u(n) := u(n+1) - u(n). \tag{2}$$

So, we obtain the logistic equation of parameter $\mu := \eta - 1$ with the initial condition rescaled by a factor $\frac{\mu+1}{\mu}$. More precisely, we have

$$\Delta u(n) = \mu u(n)(1-u(n)), \qquad u(0) = \frac{\mu+1}{\mu}v(0). \tag{3}$$

Wu and Baleanu [36] considered a fractional version of the dynamical system generated by replacing the Euler operator by the left Caputo-like discrete difference operator Δ^ν defined as follows

$$\Delta^\nu u(n) := \sum_{j=0}^{n} k^{-\nu}(n-j)u(j), \quad n \in \mathbb{N}_0, \tag{4}$$

where $k^{-\nu}(j)$ is defined by the generating series

$$\sum_{j=0}^{\infty} k^{-\nu}(j)z^j = (1-z)^\nu, \quad \nu \in \mathbb{R}. \tag{5}$$

In the last years, this definition of the difference operator Δ^ν has proven to be very useful due to its convolutional form and for being equivalent, up to translation, with others more commonly used in the literature. See [11,12,22,23] for an overview and properties. It is worth noting that for any $\nu \in \mathbb{R}$ that is not a positive integer, we have the explicit formula [3]

$$k^{-\nu}(j) = \frac{\Gamma(j-\nu)}{\Gamma(-\nu)j!}, \quad j \in \mathbb{N}_0. \tag{6}$$

See also the references [35,37] for recent works showing the applications of discrete fractional calculus. It is interesting to observe that with the given definition of Δ^ν, the fractional version of the logistic equation adopts a convolutional form and reads as follows

$$u(n) = \mu \sum_{j=1}^{n} k^\nu(n-j)u(j-1)(1-u(j-1)). \tag{7}$$

This representation gives an interpretation of the fractional version of the logistic equation as the one that incorporates a memory kernel in terms of a discrete parameter, thus incorporating a different measure of the trajectories. Given an arbitrary condition $u(0) \in [0,1]$, the trajectory $\{u(n)\}_{n=1}^N$ obtained with (7) will be called a Wu Baleanu trajectory. It is worth to mention that for $\nu = 1$ and $n = 1$ we have $\Delta u(0) = \mu u(0)(1-u(0))$, recovering (2) for $n = 0$. Fixing an initial condition and a fractional parameter ν, we can generate Feigenbaum diagrams in order to illustrate the dynamics of this dynamical system in terms of the parameter μ, see Figs. 1 and 2.

Anomalous diffusion trajectories $\{u(n)\}_{n=1}^N$ are those whose average width or variance, computed as the mean square displacement (MSD) $\langle u(n) - u(0)\rangle$ do not grow linearly respect to n, that is $\langle u^2\rangle \sim n^\nu$, with $\nu \neq 1$. The exponent ν is known as the anomalous diffusion exponent. Examples of models generating anomalous diffusion trajectories are: Annealed Transient Time Motion (ATTM) [26], Continuous-Time Random Walk (CTRW) [33], Fractional Brownian Motion (FBM) [15,25], Lévy Walks (LW) [16,29], and Scaled Brownian Motion (SBM) [20].

Recently, within the frame of the Andi Challenge [30], machine learning methods, alone or combined with some statistical measures, have demonstrated their efficiency in (i) classifying anomalous diffusion noisy trajectories according to one of the previous five generative models and (ii) inferring the anomalous diffusion exponent. We refer the reader to [29] and some subsequent works in which some of these models were fully developed [1,7,9]; see also [6,8,28]. It is also worth mentioning the recent interest in incorporating machine learning methods and intelligent algorithms in studying formal mathematical problems. We can find some examples of this approach incorporating these techniques for solving nonlinear models, such as artificial neural networks, swarm optimization, and active-set algorithms [31], or neuro-swarming heuristics [32].

Therefore, we wonder if this approach lets us infer some fractional-related trajectories' characteristics. In this work, we study if we can infer the μ parameter and

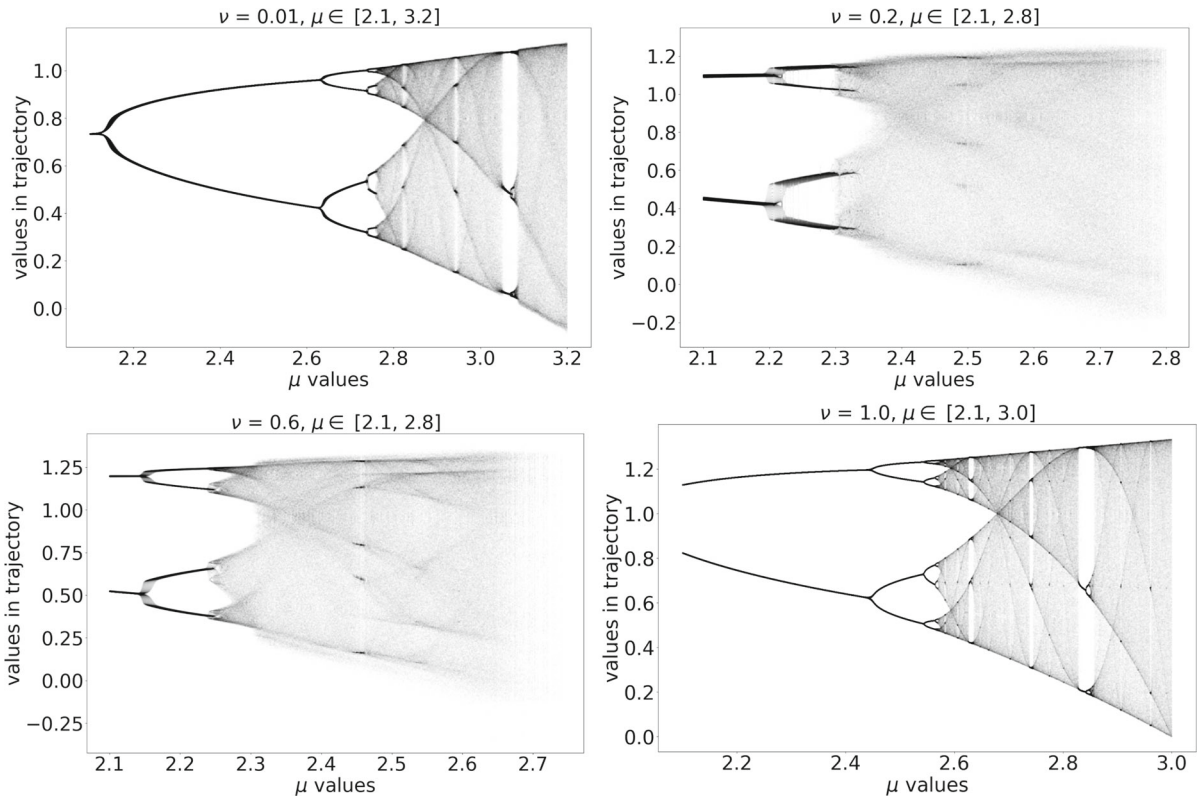

Fig. 1 Feigenbaum plots for the dynamical system given by (7) for $u_0 = 0.3$ and $\nu = 0.01$ (top left), $\nu = 0.2$ (top right), $\nu = 0.6$ (bottom left), and $\nu = 1$ (bottom right). For each value μ, we compute 200 terms of the sequence, and we plot the last 100 values

the fractional derivative order ν of Wu Baleanu trajectories. We have chosen an architecture based on recurrent neural networks (RNN), the same that provided the best results in inferring the exponent α of one-dimensional trajectories in the Andi Challenge [7,29], for trying to infer these parameters and measuring up to which point there is a straightforward relation any given trajectory of this type and the corresponding parameters μ and ν involved in generating it. To the best of our knowledge, this paper is the first to seek this type of approach. In Sect. 2, we give some details of the model architecture, revisiting some basic fundamentals of our machine learning models. We set the training, validation, and test data sets, as long as the results, in Sect. 3. Finally, we draw some conclusions in Sect. 4.

2 Architecture of the method

We propose the architecture shown in Fig. 3 to infer the parameters μ and ν of a given trajectory are introduced as input. Such architecture has been successfully applied for analyzing trajectories [7] and time series [24]. It is a mixture of convolutional and recurrent neural networks. It consists of three parts.

1. First, we have two convolutional layers that permit the extraction of spatial features from the trajectories. The first convolutional layer is set with 32 filters and a sliding window (kernel) of size 5, which slides through each trajectory extracting spatial features from them. The second convolutional layer has 64 filters to extract higher-level features.
2. Second, the output of the convolutional layers feeds three stacked bidirectional LSTMs layers that permit learning the sequential information. After each of these layers, we include a dropout layer of the 10% neurons to avoid over-fitting. We tested several dropout levels, from 5% to 20%, being 10% the one with the best performance.
3. Finally, we use two fully connected dense layers: the first one with 20 units and the second one with

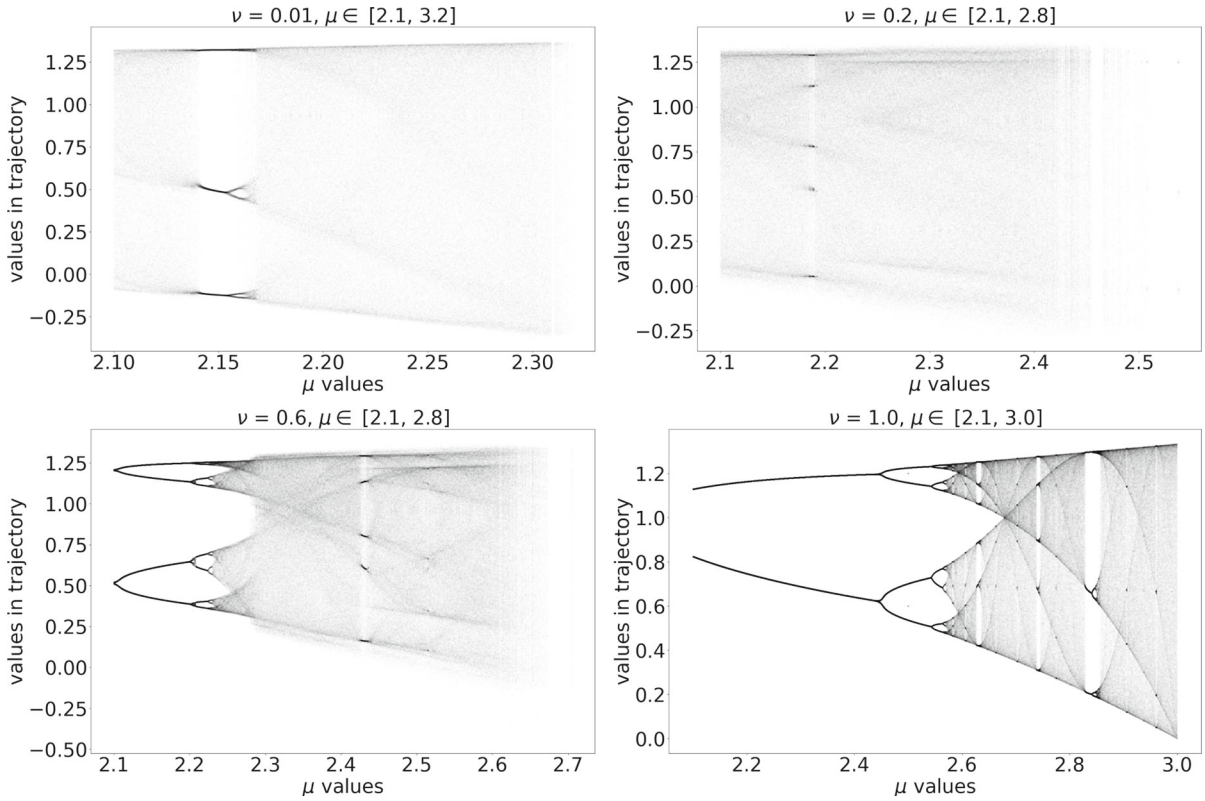

Fig. 2 Feigenbaum plots for the dynamical system given by (7) for $u_0 = 0.8$ and $\nu = 0.01$ (top left), $\nu = 0.2$ (top right), $\nu = 0.6$ (bottom left), and $\nu = 1$ (bottom right). For each value μ, we compute 200 terms of the sequence, and we plot the last 100 values

1 or 2 units. This last choice depends if we want to predict a single parameter or both of them at the same time.

Our model will be fed with trajectories of length between 10 and 50, which are the most frequent in experiments and the hardest to be classified [29]. Let us briefly describe each part of the model:

2.1 Convolutional neural networks (CNN)

Convolutional neural networks preserve the spatial structure of data. They do so by connecting a patch (or section) from data to single neurons, so every neuron learns the properties from this single patch, whose size is defined by the kernel size (5 in our model). By doing so, spatially close portions of data are likely to be related and correlated to each other since only a small region of the input data influences the output from each neuron [4,19]. The patch is slid across the input sequence, and each time we slide it, we have a new output neuron in the following layer. This lets us consider the spatial structure inherent to the input sequence [17,18]. Through these layers, we are able to learn trajectory features by weighting the connections between the patches and the neurons so that particular features can be extracted by each patch. By using multiple filters (32 and 64 in our case) the CNN layers are extracting multiple different features (linear and nonlinear) that feed our LSTM layers.

2.2 Recurrent neural networks (RNN)

Sequential information can be decomposed in single-time steps, such as words or characters in language, notes in music, codons in DNA sequences. So, if one considers sequential data, it is very likely that the output at a later time step will depend on the inputs at prior time steps. In practice, we need to relate the information from a particular time step also with prior time steps and pass this information to future times.

Fig. 3 Machine learning used for inferring the generating μ and ν parameters of Wu Baleanu trajectories

Recurrent neural networks (RNN) address this problem by adding an internal memory or cell state, denoted by h, which is passed from the time t to the time $t+1$, that is from h_t to h_{t+1}. This recurrent relation is capturing some notion of memory of what the sequence looks like. Therefore, the RNN output is not only a function of the input at a particular time step but also a function of the past memory of the cell state. In other words, the output $\overline{y_t} = f(x_t, h_{t-1})$ depends on the current input x_t and the previous inputs to the RNN h_{t-1}, as it can be seen in Fig. 4.

An RNN adapts the internal hidden state (or memory state) h_t through the result of multiplying two weight matrices W_{hh} and W_{xh} to the previous cell state h_{t-1} and the current input x_t, respectively. The weight matrix W_{hh} is modified at each time step to let the cell learn how to fit the desired output, and W_{xh} is the weight matrix that modules the contribution of the input at each time step to the learning process. The result is passed to an activation function tanh that modifies the current state at each time step, i.e., $h_t = \tanh\left(W_{hh}^T h_{t-1} + W_{xh}^T x_t\right)$.

The problem with RNNs arises when dealing with long sequences since composing multiple tanh functions entails that the hidden state tends to extinguish by reaching values very close or equal to zero. In practice, this means that only recent cell states will modify the current cell state or, in other words, that RNNs have short-term memory.

2.3 Long short-term memory (LSTM)

Long short-term memory (LSTM) [14,21] amends the aforementioned short-term memory problem implicit to RNN by including gated cells that allow them to maintain long-term dependencies in the data and to track information across multiple time steps. This improves the sequential data modeling. LSTM structure is shown in Fig. 4 where σ and tanh stand for the sigmoid and the hyperbolic tangent activation functions. The circles in red represent matrix multiplication and additions. An LSTM incorporates a new cell state channel c which can be seen as a transportation band where the info is selectively updated by the new gates and is independent of the previously defined hidden state h and, therefore, independent of what is outputted in the form of hidden state or current time step out.

One LSTM cell's composition can be seen in Fig. 4 (right), and the gates are used to control the flow of information as follows:

- The first sigmoid gate decides what information is kept or rid of. Since the sigmoid output ranges from 0 to 1, this can be seen as a switch that modulates how much information from the previous state has to be kept.
- The second gate, consisting of a sigmoid and a tanh functions store relevant information to the newly added cell state channel (c).
- Then, the outputs of the two previous gates are used to update the cell state (c) selectively.
- And the last sigmoid and tanh functions produce two different outputs; the new cell state (c), which is forwarded to the next LSTM cell, and the current time step output, which is a filtered version of the cell hidden state (h).

Further details about LSTM functioning and implementation can be found in [2,13].

3 A general model for inferring μ and ν parameters

We have built three independent data sets to infer both μ and ν simultaneously. The train, validation, and test data sets have been built with the following parameters:

- $\mu \in [2, 3.2]$ with increments of 0.001.
- $\nu \in [0.01, 1]$ with increments of 0.01.

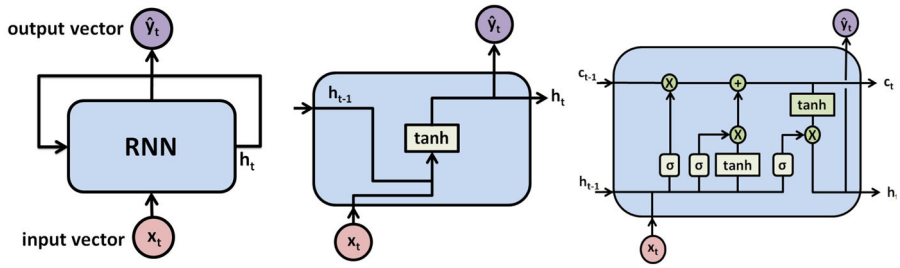

Fig. 4 Basic representation of a general RNN (left). An RNN with an inner tanh activation function (center). A scheme of an LSTM layer (right)

– trajectory length N, with $N \in [10, 50]$ randomly selected.
– $u_0 \in [0, 1]$ randomly chosen with a resolution of 10^{-2}.

We visit the μ range with higher accuracy, in order to capture the chaotic dynamics that appears in some regions, see Figs. 1 and 2. We iterate over the values of μ and ν for building the aforementioned data sets. At each iteration, per each combination of μ and ν values we randomly select 5 length value N and 5 different values of u_0, one in each one of these intervals [0.0, 0.2], (0.2, 0.4], (0.4, 0.6], (0.6, 0.8], and (0.8, 1.0], thus producing 5 trajectories of different lengths. When computing the trajectories, if we attain a value lower than -0.5 or greater than 2.0 in the trajectory, we stop the trajectory generation and save the trajectory as it is, provided it has a length greater than 10. The whole pool of trajectories is split into training (65%), validation (15%), and test (20%). As a result of this procedure, we get a training data set containing 618199 trajectories, a validation data set of 142822 trajectories, and a test data set with 190334 trajectories. The data sets can be found in supplementary material.

In all data sets, we pad each trajectory with 0's at its beginning to make them of a fixed length equal to 50. This permits homogenizing the lengths and feeding the first convolutional layer of our proposed architecture, see [10, Ch. 5 & Ch. 9]. We used an early stopping callback with a patience value of 20, which in practice means that the model stops training when validation mean average error (MAE) does not improve after 20 consecutive epochs. We have used a computer with 16 cores configured with 128 GB RAM and Nvidia RTX 3090 GPU with 22 GB RAM, running Ubuntu 20.10. The complete training process took less than 2 h, running up to 23 epochs.

First, we provide a description of the MAE distribution in terms of the true value to be predicted in Fig. 5. A point (μ, μ) in the diagonal represents that one trajectory was generated with μ as a parameter (x coordinate), and the model infers μ as the potential value used for generating the trajectory (y-coordinate). The color bar represents the density of points in the picture. Therefore, the accumulation of spots along the diagonal represents that the model predicts very well the parameters μ and ν. On the one hand, when inferring the parameter μ, the darker region along the diagonal indicates that the best results are given for medium values of μ, between 2.24 and 2.96; on the other hand, the results behave more homogeneously when predicting ν.

We point out that the results in Fig. 5 do not shed light on the importance of the trajectory length to improve the accuracy of the predictions. It is expected that the longer the trajectories are, the lower the MAE is. In order to check it, we compare the MAE results for the shortest trajectories, lengths between 10 and 19, and the longest ones, lengths between 40 and 50, in Fig. 6. We see that for long trajectories (lengths between 40 and 50), the spots concentrate along the diagonal, which indicates that the model improves its accuracy when predicting values of long trajectories. We also appreciate that in the μ case, the model improves quite a lot when predicting values of μ higher than 2.96. In contrast, in the ν case, the long trajectories show a performance improvement for values lower than 0.5.

Here, it is not easy to appreciate at first sight; however, looking at Table 1, we can see that it really holds. We can see that, except in the last case, as we increase the trajectory length, the results improve since the parameters can be better identified. In all cases, we are in MAEs of the order of 10^{-2}, that is the same order of magnitude of the ν parameter discretization, and one order less than the μ one. This justifies our choice of a thinner discretization for μ with respect to ν.

In order to look for more insightful descriptions of the MAE, for each truth value of μ and ν we have represented the quartiles Q_1, Q_2, and Q_3 of the MAE

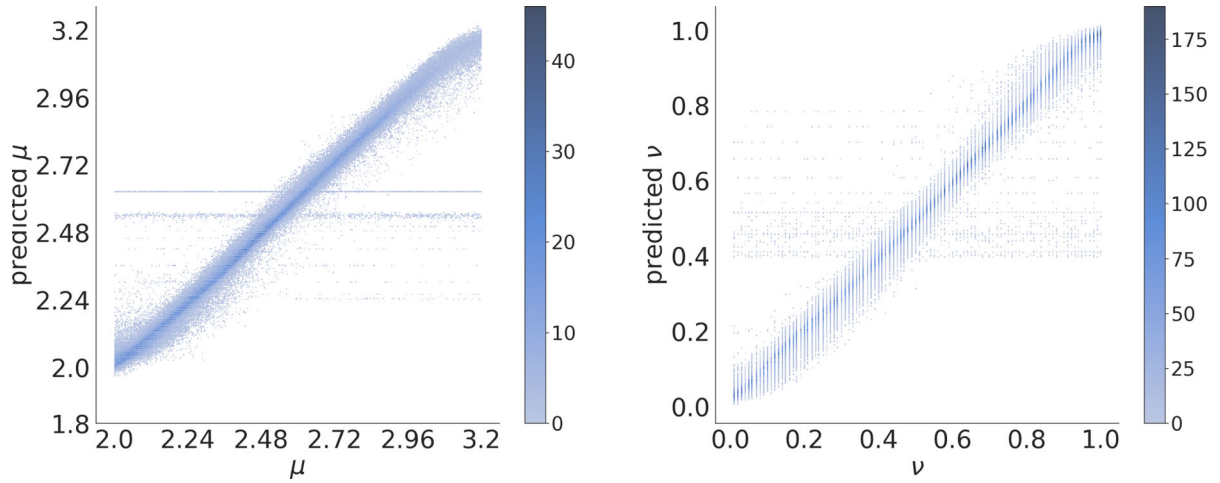

Fig. 5 Truth versus predicted values of μ and ν in the validation data set

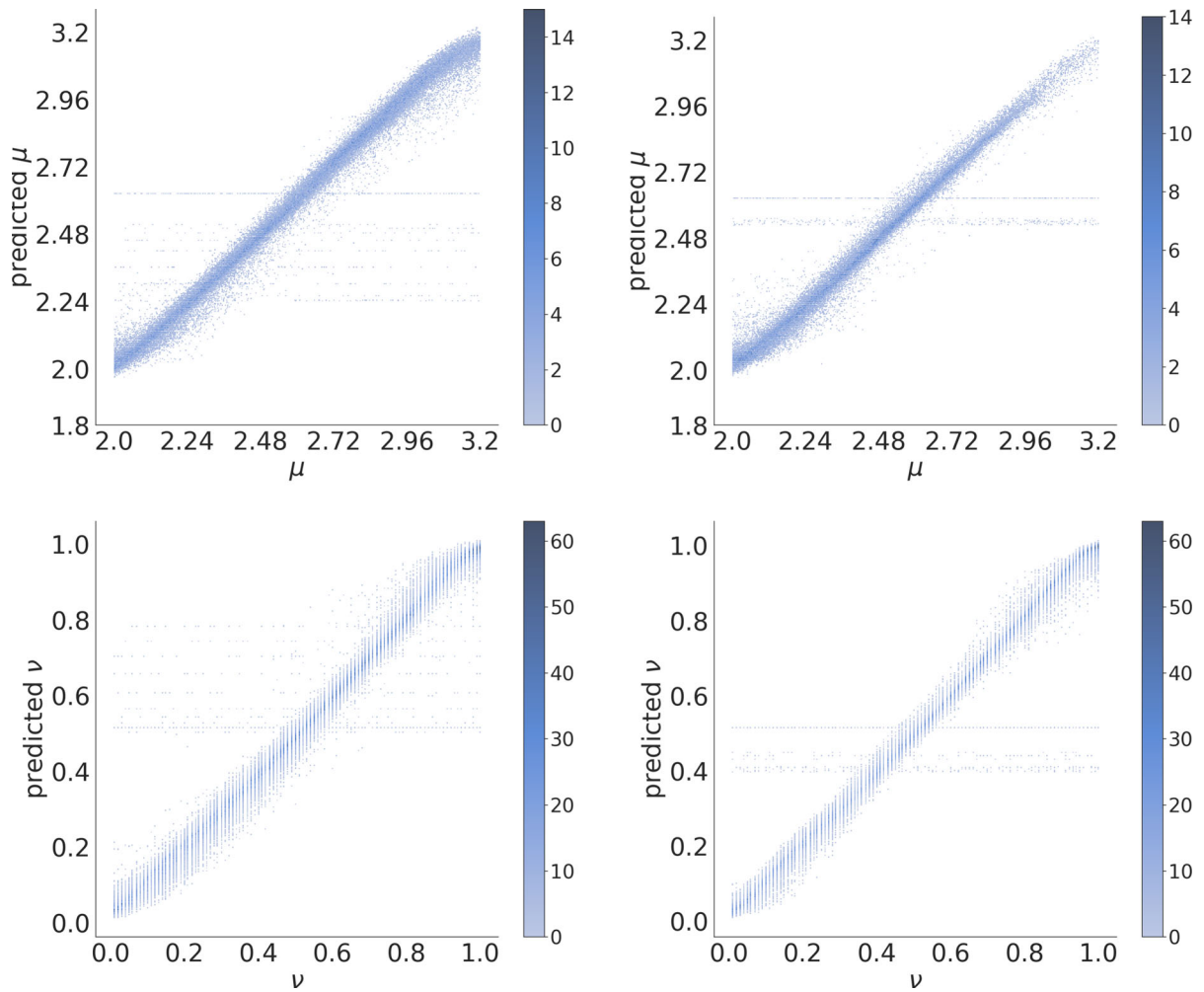

Fig. 6 Truth versus predicted values for μ and ν in the validation data sets for trajectory lengths [10–19] (left) and [40–50] (right)

Table 1 μ and ν MAE in the test data set

Length	MAE (μ)	MAE (ν)
10–19	0.0276	0.0211
20–29	0.0234	0.0173
30–39	0.0233	0.0164
40–50	0.0262	0.0186
All	0.0253	0.0186

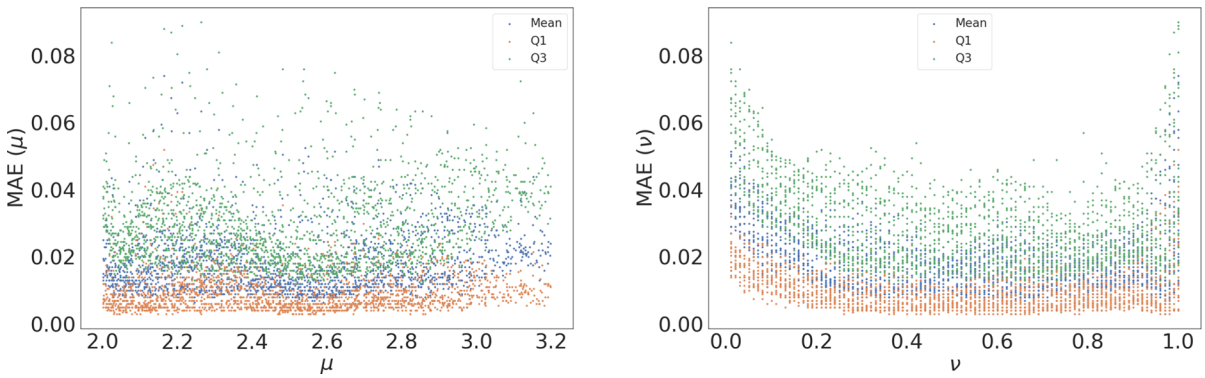

Fig. 7 Quartiles Q1 (red), Q2 (blue), and Q3 (green) of the MAE distribution on the evaluation data for μ (left) and ν (right)

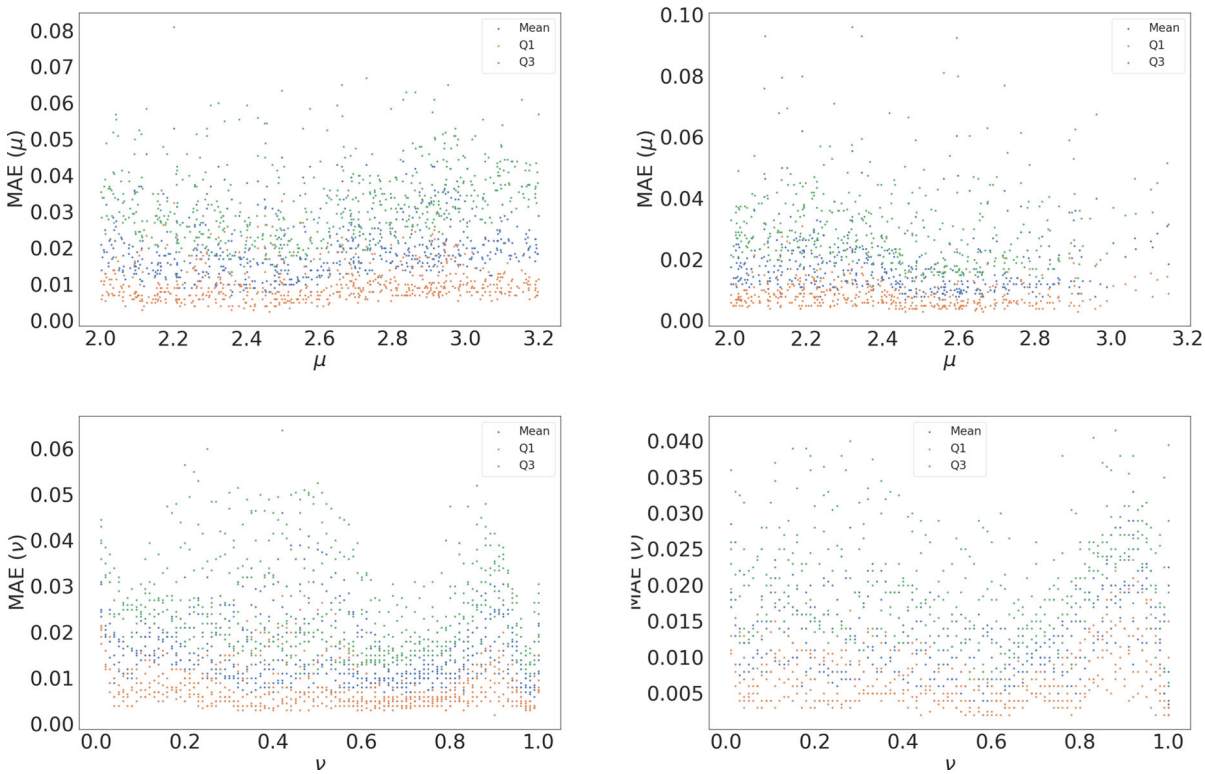

Fig. 8 Quartiles Q1 (red), Q2 (blue), and Q3 (green) of the MAE distribution on the evaluation data set associated with every single value of μ and ν for trajectory lengths [10–19] (left) and [40–50] (right)

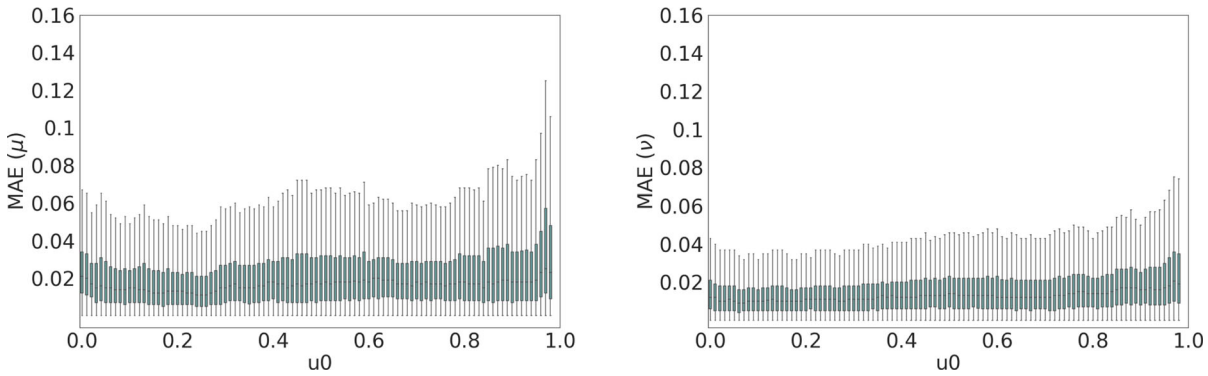

Fig. 9 MAE of μ (left) and ν (right) on the evaluation data set as a function of u_0. The darkest region coincides with the boxes, the medium gray stands for the whiskers, and the light gray for the outliers

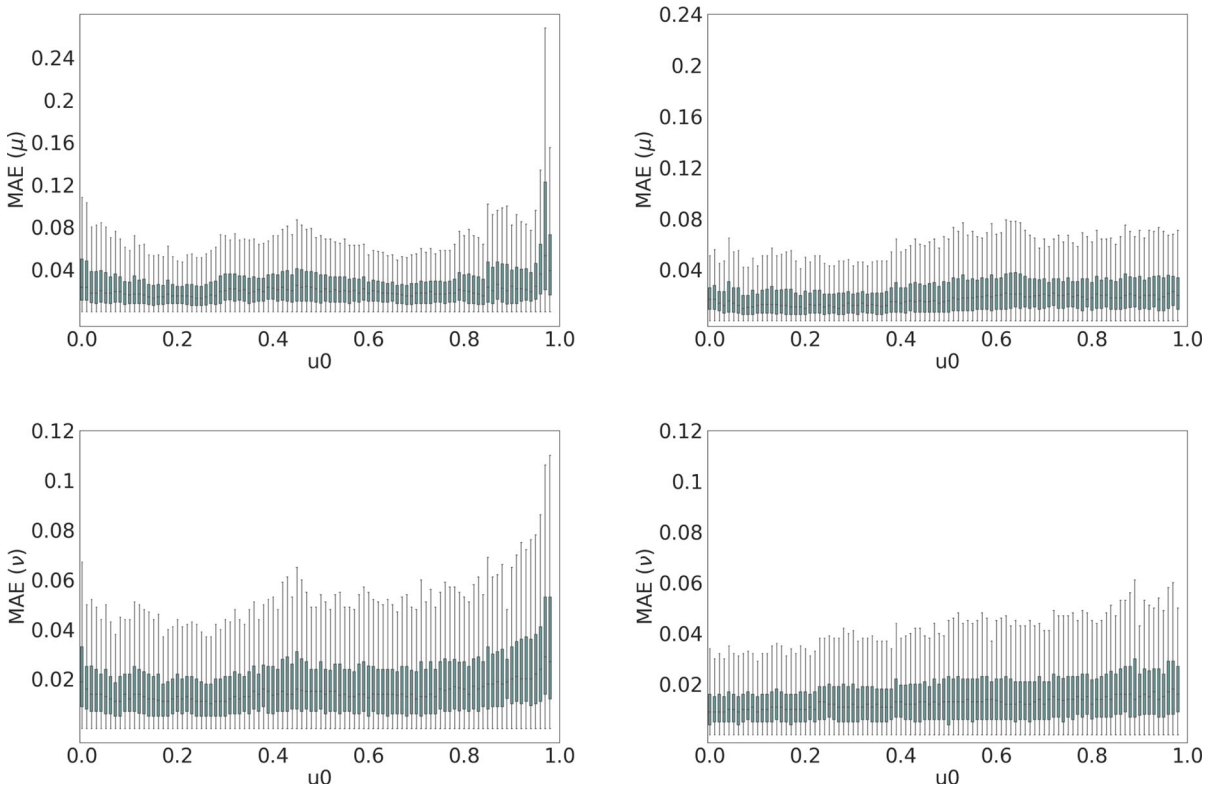

Fig. 10 MAE of μ (up) and ν (down) on the evaluation data set as a function of u_0 for trajectory lengths [10–19] (left) and [40–50] (right). The darkest region coincides with the boxes, the medium gray stands for the whiskers, and the light gray for the outliers

error distribution in Fig. 7. Looking at the quartile values for each value of μ and ν, and especially to $Q3$ (green), we can see that the predictions for μ are less accurate at the extremes for $\mu \in [2.0, 2.2] \cup [2.8, 3.2]$ and $\nu \in [0, 0.2] \cup [0.8, 1]$ than in the complementary of these sets, as we have already noticed in Fig. 5. However, we see here more clearly that the models are less accurate close to the extreme values of $\nu = 0$ and 1 due to the strong connection existing between both fractional derivative values. We provide a comparison of these MAE distributions for short and long trajectories in Fig. 8.

Due to the fractional nature of equation (7), the model has a memory component that is strongly dependent on the initial condition $u(0)$. We show boxplots of the MAE distribution in terms of $u(0)$ in Fig. 9. We can see that initial conditions are more influential for μ predictions since boxes (green) and whiskers (gray) are higher than for ν. Despite this, in both cases, the results are slightly worse for initial conditions closer to 1, due also to the nonlinear term of the logistic terms of (7). The same conclusion can be extracted when evaluating trajectories by length, as shown in Fig. 10, where we can see that the impact is of initial conditions close to 1 leads to much higher values of MAE for short trajectories, either for boxes and for whiskers.

4 Conclusions

The success of machine learning and deep learning models has arrived in almost all scientific fields. The development of mathematical proofs and arguments seems to be one of the most difficult challenges. Nevertheless, some barriers have already fallen with the discovery of new multiplication algorithms [5].

Machine learning methods can also help us in modeling tasks and in the search and fitting of parameters. In this line, we have shown these methods permit us to infer the fractional nature of a given trajectory. In particular, we have seen that such a model permits us to elucidate if, given a set of trajectories, we can propose a fractional model based on the logistic equation that would represent the underlying process with reliability. Moreover, with reasonable use of resources, we can tune the model in order to estimate the parameter of the model μ and the parameter ν of the fractional discretization, within a similar order of magnitude.

We expect that this study will foster the incorporation of machine learning tools into the study of dynamical systems and the modeling of real-life problems.

Acknowledgements We thank M.A. García-March for helpful comments and discussions on the topic.

Data availability The datasets used/analyzed during the current study are available in the supplementary material.

Declarations

Conflict of interest The authors declare no conflict of interest.

References

1. Argun, A., Volpe, G., Bo, S.: Classification, inference and segmentation of anomalous diffusion with recurrent neural networks. J. Phys. A Math. Theor. **54**(29), 294003 (2021)
2. Brownlee, J.: Long Short-Term Memory Networks with Python. eBook (2017)
3. Conejero, J., Lizama, C., Mira-Iglesias, A., Rodero, C.: Visibility graphs of fractional Wu-Baleanu time series. J. Differ. Equ. Appl. **25**(9–10), 1321–1331 (2019)
4. Cun, Y., Boser, B., Denker, J., Howard, R., Habbard, W., Jackel, L., Henderson, D.: Handwritten Digit Recognition with a Back-propagation Network, pp. 396–404. Morgan Kaufmann Publishers Inc, San Francisco (1990)
5. Fawzi, A., Balog, M., Huang, A., Hubert, T., Romera-Paredes, B., Barekatain, M., Novikov, A., Ruiz, F., Schrittwieser, J., Swirszcz, G., et al.: Discovering faster matrix multiplication algorithms with reinforcement learning. Nature **610**(7930), 47–53 (2022)
6. Firbas, N., Garibo-i-Orts, Ò., Garcia-March, M.A., Conejero, J.A.: Characterization of anomalous diffusion through convolutional transformers. J. Phys. A: Math. Theor. **56**, 014001 (2023). https://doi.org/10.1088/1751-8121/acafb3
7. Garibo-i Orts, Ò., Baeza-Bosca, A., Garcia-March, M.A., Conejero, J.A.: Efficient recurrent neural network methods for anomalously diffusing single particle short and noisy trajectories. J. Phys. A Math. Theor. **54**(50), 504002 (2021). https://doi.org/10.1088/1751-8121/ac3707
8. Garibo-i-Orts, Ò., Firbas, N., Sebastiá, L., Conejero, J.A.: Gramian angular fields for leveraging pre-trained computer vision models with anomalous diffusion trajectories. Phys. Rev. E **107**, 034138 (2023). https://doi.org/10.1103/PhysRevE.107.034138
9. Gentili, A., Volpe, G.: Characterization of anomalous diffusion classical statistics powered by deep learning (CONDOR). J. Phys. A Math. Theor. **54**(31), 314003 (2021). https://doi.org/10.1088/1751-8121/ac0c5d
10. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. Adaptive Computation and Machine Learning. MIT Press, Cambridge (2016)
11. Goodrich, G., Lizama, C.: Positivity, monotonicity and convexity for convolution operators. Discr. Cont. Dyn. Sys. A **40**(8), 4961–4983 (2020)

12. Goodrich, G., Lizama, C.: A transference principle for nonlocal operators using a convolutional approach: fractional monotonicity and convexity. Israel J. Math. **236**(2), 533–589 (2020)
13. Greff, K., Srivastava, R., Koutnik, J., Steunebrink, B., Schmidhuber, J.: LSTM: a search space odyssey. IEEE Trans. Neural Netw. Learn. Syst. **28**(10), 2222–2232 (2017)
14. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997). https://doi.org/10.1162/neco.1997.9.8.1735
15. Jeon, J., Metzler, R.: Fractional Brownian motion and motion governed by the fractional Langevin equation in confined geometries. Phys. Rev. E **81**, 021103 (2010)
16. Klafter, J., Zumofen, G.: Lévy statistics in a Hamiltonian system. Phys. Rev. E **49**, 4873–4877 (1994)
17. Krizhevsky, A., Nair, V., Hinton, G.: CIFAR-10 (Canadian Institute for Advanced Research). URL http://www.cs.toronto.edu/kriz/cifar.html**5**(4), 1 (2010)
18. Krizhevsky, A., Sutskever, I., Hinton, G.: ImageNet classification with deep convolutional neural networks. In: Proceedings of the 25th International Conference on Neural Information Processing Systems, vol. 1, NIPS'12, pp. 1097–1105. Curran Associates Inc., Red Hook (2012)
19. LeCun, Y., Bengio, Y.: Convolutional Networks for Images, Speech, and Time Series, pp. 255–258. MIT Press, Cambridge (1998)
20. Lim, S., Muniandy, S.: Self-similar Gaussian processes for modeling anomalous diffusion. Phys. Rev. E **66**(2 Pt 1), 021114 (2002)
21. Lipton, Z.: A critical review of recurrent neural networks for sequence learning. arXiv (2015). arXiv:1506.00019
22. Lizama, C.: The Poisson distribution, abstract fractional difference equations, and stability. Proc. Am. Math. Soc. **145**(9), 3809–3827 (2017)
23. Lizama, C., Murillo-Arcila, M., Peris, A.: Nonlocal operators are chaotic. Chaos **30**(10), 103126 (2020)
24. Lozano, M., Garibo-i Orts, Ò., Piñol, E., Rebollo, M., Polotskaya, K., Garcia-March, M., Conejero, J., Escolano, F., Oliver, N.: Open data science to fight COVID-19: Winning the 500k Xprize pandemic response challenge. In: Joint European Conference on Machine Learning and Knowledge Discovery in Databases, pp. 384–399. Springer (2021)
25. Mandelbrot, B., Van Ness, J.: Fractional Brownian motions, fractional noises and applications. SIAM Rev. **10**(4), 422–437 (1968). https://doi.org/10.1137/1010093
26. Massignan, P., Manzo, C., Torreño-Pina, J., Garcia-Parajo, M., Lewenstein, M., Lapeyre, G.: Nonergodic subdiffusion from Brownian motion in an inhomogeneous medium. Phys. Rev. Lett. **112**(15), 150603 (2014)
27. May, R.M., et al.: Simple mathematical models with very complicated dynamics. Nature **261**, 459–467 (1976)
28. Muñoz-Gil, G., Garcia-March, M., Manzo, C., Martín-Guerrero, J., Lewenstein, M.: Single trajectory characterization via machine learning. New J. Phys. **22**, 013010 (2020)
29. Muñoz-Gil, G., Volpe, G., Garcia-March, M.A., Aghion, E., Argun, A., Hong, C., Bland, T., Bo, S., Conejero, J.A., Firbas, N., et al.: Objective comparison of methods to decode anomalous diffusion. Nat. Commun. **12**, 6253 (2021). https://doi.org/10.1038/s41467-021-26320-w
30. Muñoz-Gil, G., Volpe, G., García-March, M., Metzler, R., Lewenstein, M., Manzo, C.: The Anomalous Diffusion challenge: objective comparison of methods to decode anomalous diffusion. In: G. Volpe, J. Pereira, D. Brunner, A. Ozcan (eds.) Emerging Topics in Artificial Intelligence (ETAI) 2021, vol. 11804, p. 1180416. Int. Soc. Opt. Photonics, SPIE (2021). https://doi.org/10.1117/12.2595716
31. Raja, M., Umar, M., Sabir, Z., Khan, J., Baleanu, D.: A new stochastic computing paradigm for the dynamics of nonlinear singular heat conduction model of the human head. Eur. Phys. J. Plus **133**(9), 1–21 (2018)
32. Sabir, Z., Raja, M., Baleanu, D., Cengiz, K., Shoaib, M.: Design of Gudermannian Neuroswarming to solve the singular Emden-Fowler nonlinear model numerically. Nonlinear Dyn. **106**(4), 3199–3214 (2021)
33. Scher, H., Montroll, E.: Anomalous transit-time dispersion in amorphous solids. Phys. Rev. B **12**, 2455–2477 (1975)
34. Strogatz, S.: Nonlinear Dynamics and Chaos: With Applications to Physics, Biology, Chemistry, and Engineering. Westview Press, Boulder (2014)
35. Wang, Z., Shiri, B., Baleanu, D.: Discrete fractional watermark technique. Front. Inf. Technol. Electron. Eng. **21**(6), 880–883 (2020)
36. Wu, G., Baleanu, D.: Discrete fractional logistic map and its chaos. Nonlinear Dyn. **1-2**, 283–287. https://doi.org/10.1007/s11071-013-1065-7
37. Wu, G., Baleanu, D.: Discrete chaos in fractional delayed logistic maps. Nonlinear Dyn. **80**(4), 1697–1703 (2015)