Int. J. Nonlinear Anal. Appl. 12 (2021) No.1, 189–201
ISSN: 2008-6822 (electronic)
http://dx.doi.org/10.22075/ijnaa.2021.4757

# Deep inference: A Convolutional Neural Networks Method for Parameter Recovery of the Fractional Dynamics

N. Biranvand[a,*], A. H. Hadian-Rasanan[b], A. Khalili[c], J. A. Rad[b]

[a]*Faculty of Sciences, Imam Ali University, Tehran, Iran*
[b]*Department of Cognitive Modeling, Institute for Cognitive and Brain Sciences, Shahid Beheshti University, Tehran, Iran*
[c]*Faculty of Engineering, Imam Ali University, Tehran Iran*

*(Communicated by Madjid Eshaghi Gordji)*

**Abstract**

Parameter recovery of dynamical systems has attracted much attention in recent years. The proposed methods for this purpose can not be used in real-time applications. Besides, little works have been done on the parameter recovery of the fractional dynamics. Therefore, in this paper, a convolutional neural network is proposed for parameter recovery of the fractional dynamics. The presented network can also estimate the uncertainty of the parameter estimation and has perfect robustness for real-time applications.

*Keywords:* Convolutional neural network, Parameter estimation, Fractional Dynamics, Data driven discovery.
*2010 MSC:* Primary 35R11; Secondary 62G07.

## 1. Introduction

Fractional dynamics have been used for modeling various phenomena in different fields including engineering, biology [1], cognitive science, and fluid dynamics[2]. So, simulating and parameter estimation of them are very important. Generally, there are three types of problem in dynamical systems:

*Corresponding author
*Email addresses:* Nabiranvand@gmail.com (N. Biranvand), amir.h.hadian@gmail.com (A. H. Hadian-Rasanan), alikhalili57@yahoo.com (A. Khalili), j.amanirad@gmail.com (J. A. Rad)

*Received:* 05-10-2020 *Revised:* 21-12-2020

- Modeling: The first level of application of fractional calculus is modeling. At the modeling level, the question is how can we interpret phenomena in terms of fractional differential equations [3].
- Simulation: In the second level, in order to study the behavior of the modeled phenomena in various conditions, it should be simulated. Therefore, the behavior of the fractional model has been simulated by different numerical algorithms.
- Parameter estimation: The last level of application for the fractional calculus is parameter estimation of appropriate fractional models obtaining from the first level. After developing a fractional model and some simulation studies on the behavior of the fractional model, it could fit on the real data sets. Thus, the existence of some parameter estimation procedures is crucial in the real-world applications of fractional dynamics. Generally, two types of problems are defined in fitting a model on the data.
  - Offline parameter estimation: In this type of fitting problem, time is not a concern and the assumption is that there is enough time for the purpose of fitting the model on data.
  - Online parameter estimation: Despite the offline problems, the online problems or real-time problems consider the computation time and the computations should be finished by a deadline. For example, in system identification problems, time of estimation is very important especially in imaging and radiography problem [4].

Therefore, if there is not exist an accurate and fast parameter recovery procedure, the fractional models can not be utilized in real-time applications. Fig (1) illustrates how these levels of applications are related to each other [5].

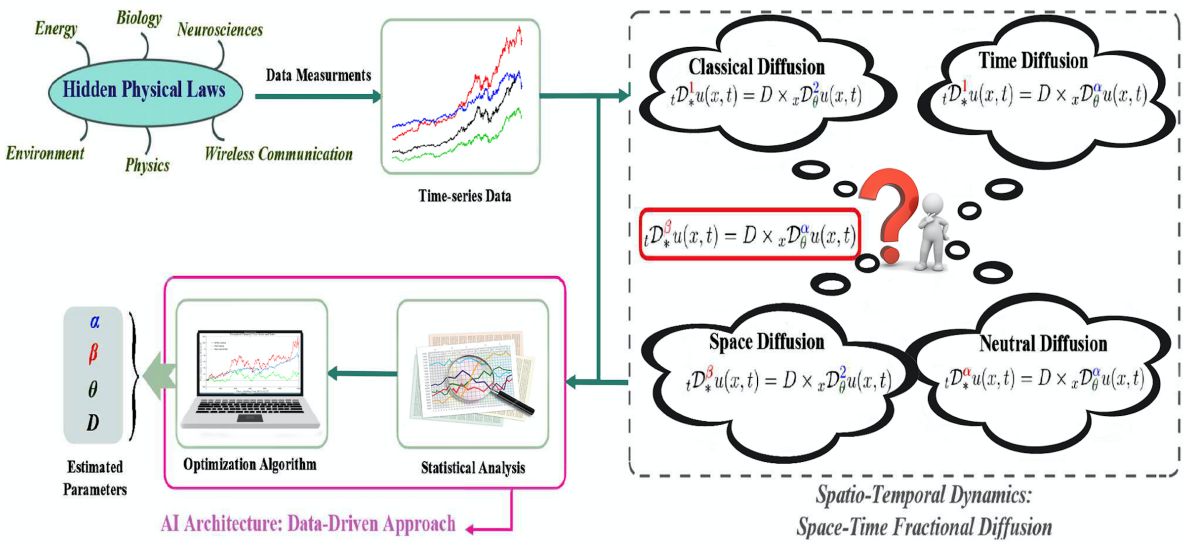


Figure 1: The flowchart of how different levels of the problem are related to each other [5].

In this paper, an online parameter estimation algorithm based on a convolutional neural network is presented. In the next section, a formal formulation for the parameter estimation of the fractional dynamics is presented.

*1.1. Statement of the problem*

Despite the simulation problems, which include a dynamic and some conditions (i.e. initial and/or boundaries), the data-driven discovery only consists of a dynamic with some unknown parameters and a set of noisy observations. So, we have:f

$$\begin{cases} K(u(x),\theta) = f(x) \\ \{(x_1,u_1),\cdots,(x_n,u_n)\} \\ x_i \in \mathbb{R}^d, u_i \in \mathbb{R} \end{cases}, \tag{1.1}$$

where $K(.,.)$ is a (non)linear fractional operator with the unknown parameters $\theta = (\theta_1,\cdots,\theta_m)$ and $\{(x_1,u_1),\cdots,(x_n,u_n)\}$ is the set of noisy observations. The aim is to find $\theta$, so the unknown parameters should be approximated in such order that the obtained dynamical system behavior be close to the actual behavior of the dynamical system in the measured points. So the problem can be reformulated as follows:

$$\theta^* = argmin_\theta \sum (U(x_i) - u_i)^2 \tag{1.2}$$

such that:

$$K(U(x),\theta) = f(x).$$

But this minimization problem is very hard to solve and the proposed algorithms for solving such optimization problems are very time-consuming. Therefore, we are going to approximate the unknown parameters of the dynamical system, but without solving the mentioned minimization problem.

*1.2. Literature review*

Until now, several works have been done on parameter estimation of the fractional dynamics. In [6], the authors have developed an algorithm based on simulating the fractional system and converting it to an optimization problem. In the first step of this algorithm, the fractional system is simulated by the fractional predictor-corrector method and then it converts to an optimization problem. After that, the optimization problem is solved with a heuristic optimization algorithm. In [7], the authors have presented an algorithm for estimation of two-dimensional space fractional models based on a combination of implicit difference method with fast bi-conjugate gradient stabilized method. Raissi and his collaborators have utilized the Gaussian process for parameter estimation of integer and fractional order linear models in [8]. They also extend this approach for the nonlinear dynamics in [9]. Recently, some neural network algorithms have been introduced by the researchers to recover the parameters of the dynamical systems. PDE-Net 2 [10], physics informed neural networks [11, 12], and DLGA-PDE [13] are some examples of neural networks that are presented for the purpose of estimating the unknown parameters.

The remaining of the paper organized in the following order: in Section 2, some preliminaries about the fractional calculus, approximate Bayesian computing, and the convolutional neural network which is used in this work are presented, in Section 3, the steps of the proposed algorithm for estimating the parameters of a fractional dynamic are discussed and in Section 4, the performance of the parameter recovery procedure is illustrated on some examples; finally, a conclusion is presented in Section 5.

## 2. Preliminaries

In this section, the preliminaries materials for the proposed algorithm is presented. These materials include, definition of the fractional derivative and basics of the fractional calculus, basics of approximate Bayesian computing, and architecture of a convolutional neural network for parameter estimation.

*2.1. Fractional derivative*

By considering $f(x)$ as a function, the Cauchy formula for n-th order can be obtained, and by generalizing the Cauchy formula for non-integer orders, we can achieve the Riemann-Liouville definition of fractional integral, and due to this, the well-known Gamma function is used as the factorial function for non-integer numbers. Hence, by denoting the Riemann-Liouville definition of the fractional order of integral as [14] $\mathcal{I}_x^\beta f(x)$, it can be defined as follows:

$$\mathcal{I}_x^\beta f(x) = \frac{1}{\Gamma(\beta)} \int_0^x (x-t)^{\beta-1} f(t)\, dt, \tag{2.1}$$

in which $\beta$ is a real number which indicates the order of integral. Moreover, there is another definition for fractional derivative, called, Caputo definition, and denoted by ${}_a^C\mathcal{D}_x^\beta f(x)$. It can be defined as follows:

$$\mathcal{D}_x^\beta f(x) = \mathcal{I}_x^{(k-\beta)} f^{(k)}(x) = \begin{cases} \frac{1}{\Gamma(k-\beta)} \int_0^x \frac{f^{(k)}(t)dt}{(x-t)^{\beta+1-k}} & \text{if } \beta \notin \mathbb{N} \\ \frac{d^\beta}{dx^\beta} & \text{if } \beta \in \mathbb{N} \end{cases}. \tag{2.2}$$

It is worth to mention that, since the fractional derivative is an integral operator, so it inheritances the linear property. Thus, we have[15]:

$$\mathcal{D}_x^\alpha(\lambda f(x) + \mu g(x)) = \lambda\ \mathcal{D}_x^\alpha f(x) + \mu\ \mathcal{D}_x^\alpha g(x) \qquad \lambda, \mu \in \mathbb{R}. \tag{2.3}$$

*2.2. Approximate Bayesian computing*

One of the powerful frameworks for the parameter estimation is the Bayesian framework. The aim of this paradigm is to maximize the conditional probability of $p(\theta|x)$ in which $\theta$ is the unknown parameter and $x$ is the recorded data. By using the Bayes rule we have [16]:

$$p(\theta|x) = \frac{p(x|\theta)\pi(\theta)}{\int p(x|\theta)d\theta}, \tag{2.4}$$

where $p(x|\theta)$ is the likelihood function and $\pi(\theta)$ is the prior distribution for the parameters. Since the value of the denominator of Bayes rule fraction is constant, to maximize $p(\theta|x)$ we can maximize the numerator of the Bayes rule fraction. So, we have:

$$p(\theta|x) \propto p(x|\theta)\pi(\theta). \tag{2.5}$$

The important thing that should be considered is that there is no exact likelihood function for the wide range of problems. Therefore, it is necessary to approximate the likelihood function [17]:

$$p(\theta|x) \propto p(x|\theta)\pi(\theta) \approx q(.|\theta)\pi(\theta). \tag{2.6}$$

After approximating the likelihood function, the main part of the approximate Bayesian computing method is constructing a reference table. The reference table contains some summary statistics of a large number of samples from the approximated posterior and it can be generated by the Algorithm (1).

**Algorithm 1** Generation of the reference table

```
procedure GENERATE TABLE(π(.), q(.))
    T ⟵ Create a blank reference table
    for i in {1, 2, ··· , N} do
        θ^(i) ⟵ sample from π(.)
        x̃^(i) ⟵ simulate from q(.|θ^(i))
        Store the pair {η(x̃^(i)), θ^(i)} in row i of the reference table T   # η(.) computes the summary statistics
    end for
    return T
end procedure
```

After obtaining the reference table, the procedure of parameter estimation is based on comparing the stored summary statistics with summary statistics of recorded data that we are going to extract its parameters. A simple algorithm based on utilizing a reference table is the rejection algorithm that compares the distance between the summary statistics of the recorded data and the stored rows of the table with a threshold [18].

### *2.3. Convolutional neural network*

Deep neural networks have obtained many achievements in solving various problems in different fields of science. Thus, it is interesting if deep neural networks can improve the precision of the parameter estimation in complex models. The idea is that the network learns the model or sampled data instead of using summary statistics. So, based on this idea, the network is considered as a function to map the sampled data to the parameter space( $\hat{\theta}^{(i)} = f_W(x^{(i)})$. So, the Gaussian conditional probability of the parameters can be defined as [19]:

$$p(\theta|\tilde{x};W) = \mathcal{N}(\theta|f_W(\tilde{x}),\sigma^2), \tag{2.7}$$

the likelihood function is obtained as:

$$p(\theta|\tilde{x};W) = \prod p(\theta^{(i)}|\tilde{x}^{(i)};W). \tag{2.8}$$

So, the negative log-likelihood function of the parameter estimation is obtained as follows:

$$\begin{aligned} -\log p(\theta|\tilde{x};W) &= -\sum \log p(\theta^{(i)}|\tilde{x}^{(i)};W) = -\sum \log \mathcal{N}(\theta^{(i)}|f_W(\tilde{x}^{(i)}),\sigma^2) \\ &= -\sum \log\left(\frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2\sigma^2}(\hat{\theta}^{(i)}-\theta^{(i)})^2}\right) = \sum \frac{\|\hat{\theta}^{(i)}-\theta^{(i)}\|}{\sigma^2} + \frac{N}{2}\log(2\pi\sigma^2). \end{aligned} \tag{2.9}$$

By minimizing the negative log likelihood function, the estimation can be done. So, the loss function of the network can be defined as:

$$\mathcal{L}(W) = \frac{1}{N}\sum \frac{\|\hat{\theta}^{(i)}-\theta^{(i)}\|}{\sigma^2} + \frac{1}{2}\log(2\pi\sigma^2). \tag{2.10}$$

This loss function is known as heteroscedastic loss. As obvious in this loss function the network should also estimate the $\sigma^2(\tilde{x^{(i)}})$. It can deduced that, the output layer of the network consists $2N$ neurons where $N$ is the number of unknown parameters. It is worth mentioning that the activation functions corresponding to unknown parameters are linear and the activation functions corresponding to the variance output are softplus function (i.e. $z = log(1 + e^x)$). On the other hand, some convolutional layers are held at the beginning of the network to extract the features of the sampled data automatically. In addition, number of the input channels of the network is $d + 1$ in which $d$ is the dimension of the dynamical system. Fig (2) shows the architecture of the convolutional neural network.

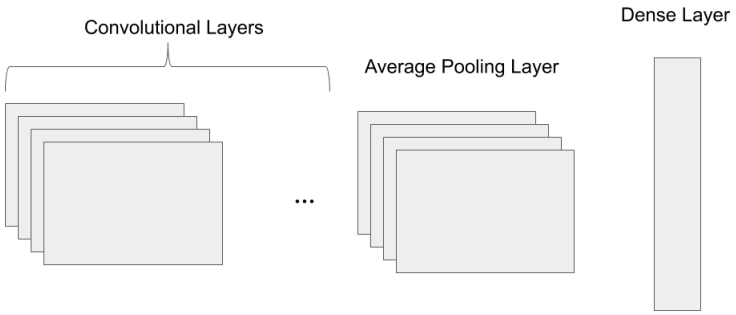


Figure 2: Structure the proposed convolutional neural network that consists some convolutional layers at the beginning, an average pooling layer and a dense layer at the end of network.

## 3. Methodology

In this section, the proposed method for recovering the unknown parameters of a fractional dynamical system is presented. The proposed algorithm consists of three steps:

i Data generation

ii Training the network

iii Prediction

The first two steps are done in an offline manner and the last step can be done in an online manner (i.e. real time). In the remaining of this section, different parts of the proposed method are discussed.

*Data generation*: The most time-consuming part of the algorithm is the data generation part. In this step, a specific model with some unknown parameters is considered. After that, based on prior knowledge about the physics of the model, some specific prior distributions are selected for the unknown parameters. Then, by generating samples from the prior distributions, a solvable model is obtained. Therefore, it is feasible to solve the obtained model numerically or analytically. There are various methods and algorithms for simulating a fractional (partial) differential equation such as multi-step methods [20], finite difference methods [21], spectral [22] and spectral element methods [23]. But all of these approaches have acceptable precision. So, there is no preference for selecting the numerical algorithm. After simulating the fractional dynamic, we should add some noise to the obtained data because the network should learn the noisy data. In the final step, the obtained noisy data set and the generated parameters should be stored. The procedure of the data simulation is presented in Algorithm (2).

**Algorithm 2** Data simulation

**procedure** GENERATE DATA($\pi(.)$, $K(.,.)$, $\sigma_{noise}$)
  T $\longleftarrow$ Create a blank reference table
  **for** $i$ in $\{1, 2, \cdots, N\}$ **do**
    $\theta^{(i)} \longleftarrow$ sample from $\pi(.)$
    $(\tilde{x}^{(i)}, \tilde{u}^{(i)}) \longleftarrow$ simulate the dynamical system $K(u(x), \theta^{(i)})$
    $e_i \longleftarrow \mathcal{N}(0, \sigma^2_{noise})$
    Store the $(\tilde{x}^{(i)}, \tilde{u}^{(i)} + e_i), \theta^{(i)}\}$ in row i of the reference table T
  **end for**
  return T
**end procedure**

*Training*: In the second step of the algorithm, we should train a neural network to learn our considered dynamical system. For this purpose, a convolutional neural network with the architecture where is mentioned in section 2.3, is utilized, and afterward, the procedure of the learning of the network is started with the obtained data set in the previous step. The network input is the noisy behavior of the system and the output of the network is the parameters of the system.

*Prediction*: In the prediction phase, a trained network is used for estimating the unknown parameters of recorded data. Since the neural network is trained and its weights are tuned, so, it can start inference by receiving the input data and there is no delay for adapting or training. Thus the in the prediction phase the proposed network can be applied to real-time problems.

## 4. Results

In this section, the performance of the proposed method is illustrated by presenting five examples. In all of these examples the fractional order the system is considered unknown and we are going to recover unknown order. The proposed examples consist two ordinary and three partial fractional differential equations. The performance of the estimation procedure is measured with the Pearson correlation between the true parameters and the estimated which is obtained by by the following formula:

$$r = \frac{n\sum \theta_i^{true}\theta_i^{estimate} - \sum \theta_i^{true}\sum \theta_i^{estimate}}{\sqrt{n\sum(\theta_i^{true})^2 - (\sum \theta_i^{true})^2}\sqrt{n\sum(\theta_i^{true})^2 - (\sum \theta_i^{true})^2}}.$$

Addition to the Pearson correlation, R-squared measure is also computed for the purpose of tracking the variance of the estimation and it can be calculated as follows:

$$R^2 = 1 - \frac{\sum(\theta_i^{true} - \frac{1}{n}\sum \theta_i^{true})^2}{\sum(\theta_i^{true} - \theta_i^{estimate})^2}.$$

All the examples have use a network which has five convolutional layers with the 64, 64, 128, 128, 128 filter sizes respectively.

### *4.1. Test example 1*

As the first example, a nonlinear fractional differential equation is considered as follows [22]:

$$D_t^{\alpha} y(t) = \frac{40320}{\Gamma(9-\alpha)}x^{8-\alpha} - 3\frac{\Gamma(5+\frac{\alpha}{2})}{\Gamma(5-\frac{\alpha}{2})}x^{4-\frac{\alpha}{2}} + \frac{9}{4}\Gamma(\alpha+1) + \left(\frac{3}{2}x^{\frac{\alpha}{2}} - x^4\right)^3 - y(t)^{\frac{3}{2}} \qquad 0 < \alpha < 2. \quad (4.1)$$

The training data set of this problem consists 24750 noisy samples with the normal noise $\mathcal{N}(0, 0.2)$. These samples simulate the behavior of the Eq (4.1) in $t \in [0, 2]$ and $\alpha \in [0, 2]$. The Pearson correlation of the estimated parameters and the true parameters is 0.99 which is very high value for the correlation. Moreover, the R-squared value is 0.98 which yields the variance of estimation is very low and has good precision. Fig (3) shows the scatter plot of the recovered parameters with different value for $\alpha$.

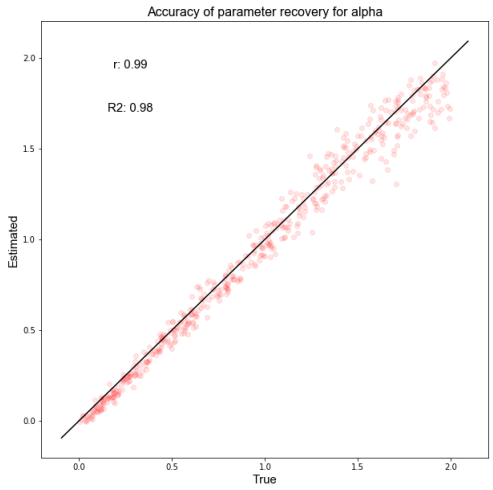


Figure 3: Quality of estimating $\alpha$ parameter for example 4.1

It is clear that the network can recover the low values of the $\alpha$ better than the high values of the $\alpha$ for this problem. The precision of parameter estimation can be effected by number of training point. In Table (1), the obtained $r$, and $R^2$ for estimating, after training with 14850, 19800, and 24750 are presented. These results yield that, we can obtain enough precision for parameter estimating with less number of training points.

Table 1: Evaluation of the performance of the proposed algorithm with different numbers of training points in example 4.1

| | #training points | | |
|---|---|---|---|
| | 14850 | 19800 | 24750 |
| $r$ | 0.99 | 0.99 | 0.99 |
| $R^2$ | 0.96 | 0.98 | 0.98 |

### *4.2. Test example 2*

For the second example, the following linear fractional differential equation is considered [22]:

$$\mathcal{D}_t^{\alpha} y(t) + y(t) = 0 \qquad 0 < \alpha < 2. \tag{4.2}$$

The training data set of this problem consists of 24750 noisy samples with the normal noise $\mathcal{N}(0, 0.2)$. These samples simulate the behavior of the Eq (4.2) in $t \in [0, 2]$ and $\alpha \in [0, 2]$. Similar results have repeated for this example. The Pearson correlation value is equal to 0.99 and the R-squared value is equal 0.97. Fig (4) presents the scatter plot of the estimated parameters with different value for $\alpha$.

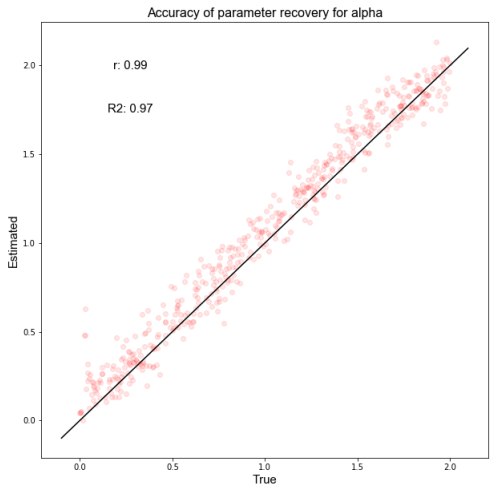


Figure 4: Quality of estimating $\alpha$ parameter for example 4.2

The variance of estimation is approximately constant for all values of the $\alpha$ parameter. Similar to previous example, the evaluation of the performance of the introduced algorithm with different number of training data is presented in Table (2) and similar result is repeated again.

Table 2: Evaluation of the performance of the proposed algorithm with different numbers of training points in example 4.2

| | #training points | | |
|---|---|---|---|
| | 14850 | 19800 | 24750 |
| $r$ | 0.98 | 0.98 | 0.99 |
| $R^2$ | 0.96 | 0.96 | 0.97 |

### *4.3. Test example 3*

The third example is a linear non-homogeneous time fractional diffusion equation [24]. Parameter recovery of this type of equation is very important because of applications of this family in source identification.

$$\mathcal{D}_t^{\alpha} u(x,t) + u(x,t) = \frac{\partial^2 u(x,t)}{\partial x^2} + \frac{2t^{2-\alpha}}{\Gamma(3-\alpha)} x(2-x) + x(2-x)t^2 + 2t^2. \tag{4.3}$$

The convolutional layers of the network for this problem has three channels. Moreover, 49500 noisy sample have been used for training the network and the domain of simulation is $(x,t) \in [0,2] \times [0,2]$ and the domain of simulation for $\alpha$ is $[0.05, 1]$. Additionally, the noise which is added to simulation data has normal distribution with mean 0 and variance 0.2. The correlation value and the R-squared value are 0.99, and 0.98 yielding the power of algorithm in recovering the of multi-dimensional models.

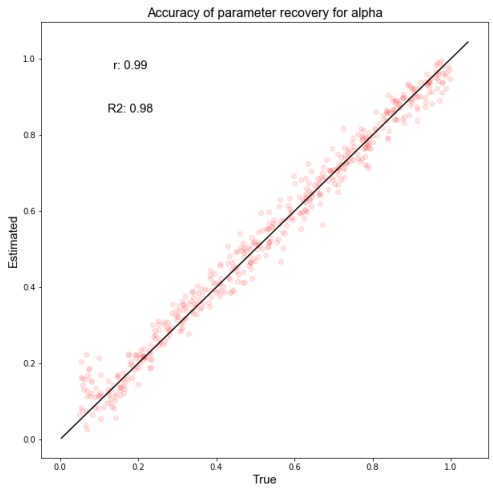


Figure 5: Quality of estimating $\alpha$ parameter for example 4.3

Table (3) which compares the performance of the proposed method with various amount of training data shows that the proposed method have enough good precision in estimating the unknown parameters by different numbers of training point.

Table 3: Evaluation of the performance of the proposed algorithm with different numbers of training points in example 4.3

| | #training points | | |
|---|---|---|---|
| | 29700 | 39600 | 49500 |
| $r$ | 0.99 | 1 | 0.99 |
| $R^2$ | 0.97 | 1 | 0.96 |

### *4.4. Test example 4*

The fourth example is a time fractional nonlinear Fokker-Plank equation [25]. This problem arise in fitting a probability distribution on data and have many applications in various field s of science. Consider the following equation:

$$\mathcal{D}_t^{\alpha} u(x,t) = [\frac{\partial}{\partial x}(\frac{4u(x,t)}{x} - \frac{x}{3}) + \frac{\partial^2 u(x,t)}{\partial x^2}]u(x,t). \tag{4.4}$$

For this problem 49500 noisy samples with $N(0, 0.2)$ noise are utilized. Moreover, the domain of the problem is considered $(x,t) \in [0,1] \times [0,1]$, and the $\alpha$ parameter located in the interval $[0,1]$ for the simulations. The quality of the recovery is presented in Fig (6).

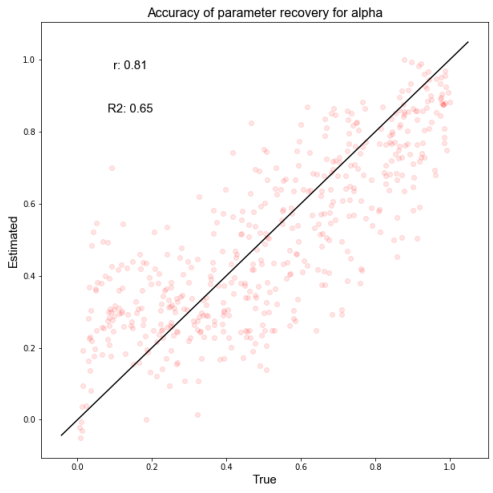


Figure 6: Quality of estimating $\alpha$ parameter for example 4.4

The obtained correlation value and R-squared value is equal to 0.81 and 0.65, respectively. It is true that the precision of the estimation for this example is not very high but it is acceptable because the value of R-squared is greater than $\frac{1}{2}$. Moreover, the effect of number of training points on the performance of the proposed neural network is presented in Table (4).

Table 4: Evaluation of the performance of the proposed algorithm with different numbers of training points in example 4.4

| | #training points | | |
|---|---|---|---|
| | 29700 | 39600 | 49500 |
| $r$ | 0.8 | 0.81 | 0.81 |
| $R^2$ | 0.64 | 0.65 | 0.65 |

### *4.5. Test example 5*

The example is a homogeneous time fractional 3-dimensional heat equation as below [26]:

$$\mathcal{D}_t^{\alpha} u(x,y,t) = \frac{\partial^2 u(x,y,t)}{\partial x^2} + \frac{\partial^2 u(x,y,t)}{\partial y^2}. \tag{4.5}$$

Since this example is a 3-dimensional problem, so the training network should have 4 channels for the input convolutional layer. 74250 noisy samples representing the behaviour of the mention heat equation in the domain $(x,y,t) \in [0,1] \times [0,1] \times [0,1]$ and for $\alpha \in [0.15,1]$, have been used to train the network. The added noise is $N(0,0.1)$ for this example. The correlation value is equal to 0.96 and the R-squared value is equal to 0.92. These values imply that the recovery procedure works well for this problem. Fig (7) shows the quality of the parameter recovery for this example.

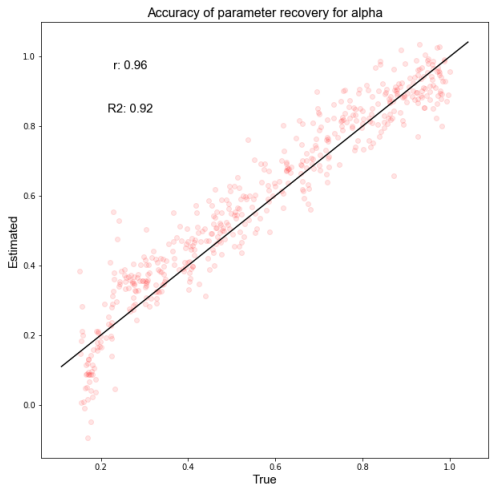


Figure 7: Quality of estimating $\alpha$ parameter for example 4.5

Generally the precision of the parameter estimation is good but by decreasing the value of alpha, the variance of estimation increases. Table (5) shows the performance of the proposed method by various numbers of training data.

Table 5: Evaluation of the performance of the proposed algorithm with different numbers of training points in example 4.5

| | #training points | | |
|---|---|---|---|
| | 49500 | 39600 | 74250 |
| $r$ | 0.97 | 0.96 | 0.96 |
| $R^2$ | 0.94 | 0.91 | 0.92 |

## 5. Conclusion

In this paper, a convolutional neural network with heteroscedastic loss function has been used for the purpose of parameter estimation of the fractional dynamics. The proposed method can be applied to real-time applications. Besides, it has acceptable accuracy for recovering unknown parameters from noisy data and it can handle both linear and nonlinear problems. Additionally, one of the main advantages of the proposed method is that the network can handle the dimension of the model by adding a channel to the convolutional layers. The accuracy of the recovery procedure, for the nonlinear cases, is not as well as the linear ones but it is acceptable. Moreover, by using more layers in the network or more complex architectures it is possible to improve the accuracy of the estimation while the network training time increases.

## References

[1] Dumitru Baleanu and António Mendes Lopes. *Handbook of Fractional Calculus with Applications*. De Gruyter, 2019.

[2] Vladimir V Kulish and José L Lage. Application of fractional calculus to fluid mechanics. *J. Fluids Eng.*, 124(3):803–806, 2002.

[3] Nasser Hassan Sweilam, Seham Mahyoub Al-Mekhlafi, Taghreed Assiri, and Abdon Atangana. Optimal control for cancer treatment mathematical model using Atangana–Baleanu–Caputo fractional derivative. *Advances in Difference Equations*, 2020(1):1–21, 2020.

[4] Bo Ni and Huajian Gao. A deep learning approach to the inverse problem of modulus identification in elasticity. *MRS Bulletin*, pages 1–7, 2020.

[5] Mohamed Ridha Znaidi, Gaurav Gupta, Kamiar Asgari, and Paul Bogdan. Identifying arguments of space-time fractional diffusion: data-driven approach. *Front. Appl. Math. Stat. 6: 14. doi: 10.3389/fams*, 2020.

[6] Fawang Liu and Kevin Burrage. Novel techniques in parameter estimation for fractional dynamical models arising from biological systems. *Computers & Mathematics with Applications*, 62(3):822–833, 2011.

[7] Shanzhen Chen, Fawang Liu, Xiaoyun Jiang, Ian Turner, and Kevin Burrage. Fast finite difference approximation for identifying parameters in a two-dimensional space-fractional nonlocal model with variable diffusivity coefficients. *SIAM Journal on Numerical Analysis*, 54(2):606–624, 2016.

[8] Maziar Raissi, Paris Perdikaris, and George Em Karniadakis. Machine learning of linear differential equations using gaussian processes. *Journal of Computational Physics*, 348:683–693, 2017.

[9] Maziar Raissi, Paris Perdikaris, and George Em Karniadakis. Numerical Gaussian processes for time-dependent and nonlinear partial differential equations. *SIAM Journal on Scientific Computing*, 40(1):A172–A198, 2018.

[10] Zichao Long, Yiping Lu, and Bin Dong. PDE-Net 2.0: Learning PDEs from data with a numeric-symbolic hybrid deep network. *Journal of Computational Physics*, 399:108925, 2019.

[11] Maziar Raissi, Paris Perdikaris, and George E Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019.

[12] Guofei Pang, Lu Lu, and George Em Karniadakis. fPINNs: Fractional physics-informed neural networks. *SIAM Journal on Scientific Computing*, 41(4):A2603–A2626, 2019.

[13] Hao Xu, Haibin Chang, and Dongxiao Zhang. DLGA-PDE: Discovery of PDEs with incomplete candidate library via combination of deep learning and genetic algorithm. *Journal of Computational Physics*, page 109584, 2020.

[14] Amir Hosein Hadian-Rasanan, Dara Rahmati, Saeide Gorgin, and Kourosh Parand. A single layer fractional orthogonal neural network for solving various types of Lane–Emden equation. *New Astronomy*, 75:101307, 2020.

[15] Amir Hosein Hadian Rasanan, Nastaran Bajalan, Kourosh Parand, and Jamal Amani Rad. Simulation of nonlinear fractional dynamics arising in the modeling of cognitive decision making using a new fractional neural network. *Mathematical Methods in the Applied Sciences*, 43(3):1437–1466, 2020.

[16] Andrew Gelman, John B Carlin, Hal S Stern, David B Dunson, Aki Vehtari, and Donald B Rubin. *Bayesian data analysis*. CRC press, 2013.

[17] Brandon M Turner and Trisha Van Zandt. A tutorial on approximate Bayesian computation. *Journal of Mathematical Psychology*, 56(2):69–85, 2012.

[18] Donald B Rubin. Bayesianly justifiable and relevant frequency calculations for the applies statistician. *The Annals of Statistics*, pages 1151–1172, 1984.

[19] Stefan T Radev, Ulf K Mertens, Andreas Voss, and Ullrich Köthe. Towards end-to-end likelihood-free inference with convolutional neural networks. *British Journal of Mathematical and Statistical Psychology*, 73(1):23–43, 2020.

[20] Kai Diethelm, Neville J Ford, and Alan D Freed. Detailed error analysis for a fractional adams method. *Numerical algorithms*, 36(1):31–52, 2004.

[21] Mostafa Abbaszadeh and Mehdi Dehghan. Fourth-order alternating direction implicit (adi) difference scheme to simulate the space-time riesz tempered fractional diffusion equation. *International Journal of Computer Mathematics*, pages 1–24, 2020.

[22] Abbas Saadatmandi and Mehdi Dehghan. A new operational matrix for solving fractional-order differential equations. *Computers & mathematics with applications*, 59(3):1326–1336, 2010.

[23] Mostafa Abbaszadeh and Hanieh Amjadian. Second-order finite difference/spectral element formulation for solving the fractional advection-diffusion equation. *Communications on Applied Mathematics and Computation*, pages 1–17, 2020.

[24] Mehmet Yavuz and Necati Özdemir. Numerical inverse Laplace homotopy technique for fractional heat equations. *Thermal Science*, 22:185–194, 2018.

[25] Jafar Eshaghi, Hojatollah Adibi, and Saeed Kazem. On a numerical investigation of the time fractional Fokker–Planck equation via local discontinuous Galerkin method. *International Journal of Computer Mathematics*, 94(9):1916–1942, 2017.

[26] Saeed Kazem and Mehdi Dehghan. Semi-analytical solution for time-fractional diffusion equation based on finite difference method of lines (MOL). *Engineering with Computers*, 35(1):229–241, 2019.