Enhanced Parameter Estimation in the Generalized Kuramoto-Sivashinsky Equation Using Physics-Informed Neural Networks and Maximum Likelihood Estimation

Mahya Pashapour, Mostafa Abbaszadeh, Mehdi Dehghan*

Abstract

When modeling phenomena with Ordinary Differential Equations (ODEs) or Partial Differential Equations (PDEs), ensuring the accuracy of numerical simulation results is crucial. The parameters of a model significantly influence its output and, consequently, its accuracy. One effective approach to improving numerical solutions' precision is parameter estimation. Therefore, a reliable tool for parameter estimation is essential. We perform parameter estimation on the Generalized Kuramoto-Sivashinsky (GKS) equation using statistical methods based on Maximum Likelihood Estimation (MLE) and machine learning techniques that employ Physics-Informed Neural Networks (PINNs). This equation is a fourth-order nonlinear PDE and has different parameters. We compare the results obtained from both approaches, demonstrating that the outcomes achieved with PINNs significantly surpass the accuracy of those obtained using MLE.

Keywords: Partial Differential Equations, Ordinary Differential Equation, Generalised Kuramoto-Sivashinsky equation, Maximum Likelihood Estimation, and Physics-Informed Neural network

1 Introduction

Many equations used to model physical phenomena do not have analytical solutions, or when such solutions exist, they can be computationally expensive to evaluate [1]. As a result, these equations are often addressed using numerical methods such as finite differences, finite elements, finite volumes, and boundary elements. However, since these methods depend on numerical approximations for derivatives or integrals to compute various terms in the equations, the solutions derived may deviate from experimental data. In addition to the numerical solution of the equation, we must also pay attention to the accuracy of the result.
Models typically contain parameters that are tied to the underlying mechanisms of the phenomena being studied [13]. As the number of parameters increases, so does the complexity of the model. These parameters can significantly influence the model's output. A common strategy for enhancing the accuracy of simulation results is to adjust the values of parameters to minimize the error—a process known as parameter estimation. By comparing the model's output with experimental data, values of parameters can be optimized, leading to improved accuracy.
One notable equation widely utilized today is the Generalized Kuramoto–Sivashinsky (GKS) equation, which is applied in plasma studies [3]. Analytical solutions to this equation can be derived for certain specific parameter values. In this study, we will conduct parameter estimation for the GKS equation using both statistical methods and machine learning techniques, followed by a comparison of the results obtained from these two approaches.

1.1 Main Model

One of the most important PDEs in the field of studying phase turbulence and unstable processes [2, 3] is the Generalised Kuramoto-Sivashinsky equation in the above form:

$$\frac{\partial u}{\partial t}+\frac{\partial u}{\partial x}+u\frac{\partial u}{\partial x}+\alpha\frac{\partial^2 u}{\partial x^2}+\beta\frac{\partial^3 u}{\partial x^3}+\gamma\frac{\partial^4 u}{\partial x^4}=0. \quad (1)$$

Some applications of this equation are working on unstable drift waves in plasma, the flame front instability [4], and long waves on thin films [5]. In this equation, α, β, and γ are non-zero parameters.
When $\alpha = 1$, $\beta = 4$, and $\gamma = 1$ we have the above analytical solution:

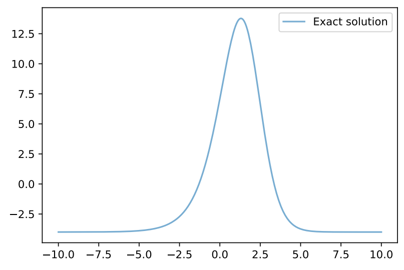
$$u(x,t) = 11+15tanh(\theta)-15tanh^2(\theta)-15tanh^3(\theta), \quad (2)$$

subject that $\theta = -\frac{1}{2}x+t$. And also, when $\alpha = 2$, $\beta = 0$, and $\gamma = 1$ we have the above analytical solution:

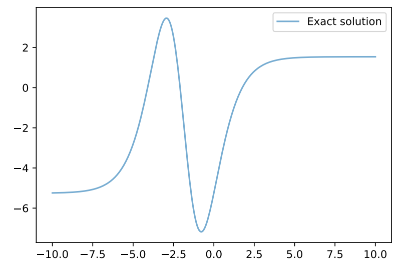
$$u(x,t) = -\frac{1}{\kappa}+\frac{60}{19}\kappa(-38\gamma\kappa^2+\alpha)tanh(\theta)+120\gamma\kappa^3tanh^3(\theta), \quad (3)$$

subject that $\kappa = \frac{1}{2}\sqrt{\frac{11\alpha}{19\gamma}}$, and $\theta = \kappa x + t$. As we have the exact values of parameters for Eq. (2) and 3 we can compare the results of parameter estimation obtained from PINNs and MLE.

*Department of Mathematics and Computer Science, Amirkabir University of Technology, `mahyapashapour@aut.ac.ir`, `m.abbaszadeh@aut.ac.ir`, `mdehghan@aut.ac.ir`



(a)



(b)

Figure 1: Image 1a and 1b corresponds to the plot of Eq. (2) and 3 respectively. The domain of x for both of equations is (-10,10) and the final time is $T = 1s$.

1.2 Maximum Likelihood Estimation

Let x_1,...,x_N be N independent and identically distributed (i.i.d.) random samples, and let $f_X(x)$ be the probability density function of these random samples. In this case, the likelihood function is given as follows [6]:

$$L(\theta) = f_{X_1,X_2...,X_N}(x_1, x_2.., x_n) = \prod_{i=1}^{N} f_X(x_i) \qquad (4)$$

where θ is unknown parameter. In the parameter estimation process, we begin by computing the likelihood function based on the provided samples. The objective is to find the value of θ that maximizes this likelihood function. To simplify the calculations, θ is often estimated by minimizing the negative logarithm of the likelihood function [6]. In addition to parameter estimation, Equation 4 can be utilized to identify identifiable parameters through the use of profile likelihood functions [7].
While this method employs probabilistic techniques, it also presents several challenges. It is sensitive to the initial values of the parameters, meaning that even minor adjustments can lead to significant variations in the results [8]. Moreover, when using this method, the physical conditions of the problem may not be adequately integrated into the likelihood function, resulting in a notable discrepancy between the model's output and experimental data [8]. Additionally, this approach may yield local optima instead of global ones for the estimated values [8].

1.3 Physics-Informed Neural Network

In recent years, many ODEs and PDEs have been solved by using Deep Neural Networks (DNN) [9]. In this method, the physical laws of the model are considered during the learning process. [10] utilized Artificial Neural Networks (ANNs) to solve ODEs and some PDEs. [11] introduced PINNs to solve PDEs and estimate parameters in the context of forward problems and inverse problems respectively.
In forward problems by PINNs we suppose the above PDE as:

$$u_t + \mathcal{N}_x[u] = 0, \quad x \in \Omega, \quad t \in (0, T], \qquad (5)$$

$$u(0, x) = h(x), \quad x \in \Omega, \qquad (6)$$

$$u(x, t) = g(t, x), \quad x \in \partial\Omega, \quad t \in (0, T], \qquad (7)$$

In this problem $\Omega \subset \mathbb{R}^D$ and $\mathcal{N}_x[u]$ is a linear or nonlinear operator, parameterized by λ. The functions h and g denote the initial and boundary conditions respectively. To gain an approximation solution, a Fully Connected Neural Network (FCNN) is used where the inputs are (t, x) and the output is $u_{NN}(t, x)$ [14]. Assume data points N_0 and N_b, which satisfy initial and boundary conditions, respectively. N_r is the number of interior points (collocation points). To gain the approximate solution, the loss function of the neural network is defined as:

$$L = L_0 + L_b + L_r, \qquad (8)$$

such that

$$L_0 = \frac{1}{N_0} \sum_{i=1}^{N_0} |(u(t_i, x_i) - h(x_i))|^2 , \qquad (9)$$

$$L_b = \frac{1}{N_b} \sum_{i=1}^{N_b} |(u(t_i, x_i) - g(t_i, x_i))|^2 , \qquad (10)$$

$$L_r = \frac{1}{N_r} \sum_{i=1}^{N_r} |u_{t_i} + \mathcal{N}_x[u_i]|^2 . \qquad (11)$$

The terms L_0 and L_b represent satisfying $u_{NN}(t, x)$ in initial and boundary conditions, respectively, while L_r corresponds to the residual of the PDE. In the process of learning, Automatic Differentiation is used to perform the loss function [15].
In parameter estimation we have: $u_t + \mathcal{N}_x[u; \lambda] = 0$. Here N_d is the number of data points within the domain. For this problem the definition of loss function is as follows:

$$L = L_d + L_r, \qquad (12)$$

where

$$L_d = \frac{1}{N_d}\sum_{i=1}^{N_d} \left|(u(t_i,x_i) - u_{NN}(t_i,x_i))\right|^2, \qquad (13)$$

$$L_r = \frac{1}{N_d}\sum_{i=1}^{N_d} \left|u_{t_i} + \mathcal{N}_x[u_i;\lambda]\right|^2. \qquad (14)$$

In this case, the parameters of the problem are learned by minimizing the loss function.
In Section 2 we will use MLE and PINN methods to estimate the parameters of Eq. (1). Section 3 represents the results obtained by two methods and also comparison between them. In section 4 we have the conclusion of this work.

2 Methodology

For parameter estimation by MLE and PINN, we have to set initialize parameter values. For Eq. (2) we set $\alpha = 5$, $\beta = 3$, and $\gamma = 4$. Also for Eq. (3) we set $\alpha = 5$, $\beta = 2.5$, and $\gamma = 4$. As we have the exact solution there is no need to calculate the numerical solution of GKS Eq. Instead of that we produce noisy data in the above way:

$$U_{noisy}(x,t) = U_{exact}(x,t) + \epsilon, \quad \epsilon \sim \mathcal{N}(0,\sigma^2). \qquad (15)$$

Next, we compute the derivatives of the two equations with respect to the inputs xx and tt to derive the structure of Equation (1). By defining the likelihood function based on the normal distribution and employing a suitable minimization algorithm for the negative logarithm of the likelihood function, we obtain the estimated parameter values.
In PINN we use the exact noisy data, and define loss function as:

$$L = \lambda L_d + L_r, \qquad (16)$$

$$L_d = \frac{1}{N_d}\sum_{i=1}^{N_d} \left|(U_{noisy}(t_i,x_i) - U_{NN}(t_i,x_i))\right|^2, \qquad (17)$$

where

$$L_r = \frac{1}{N_d}\sum_{i=1}^{N_d} \left|\frac{\partial U}{\partial t} + \frac{\partial U}{\partial x} + u\frac{\partial U}{\partial x} + \alpha\frac{\partial^2 U}{\partial x^2} + \beta\frac{\partial^3 U}{\partial x^3} + \gamma\frac{\partial^4 U}{\partial x^4}\right|^2. \qquad (18)$$

We employed a Fully Connected Neural Network (FCNN) consisting of 300 data points, three hidden layers, and 30 neurons in each layer, utilizing the tanh activation function. This implementation was carried out using PyTorch [12]. The neural network learns the parameters specified in Eq. (18). The hyperparameter λ in Eq. (16) balances the terms of the loss function.

3 Results

In this section, we have the parameter estimation results obtained by MLE and PINNs. In Tables 1 and 2 we can see the values obtained by both methods.
The results indicate that the accuracy of the estimates obtained using PINNs is remarkably high, with the parameter values converging closely to their true values. In contrast, the estimates derived from MLE show considerable discrepancies from the actual parameter values.

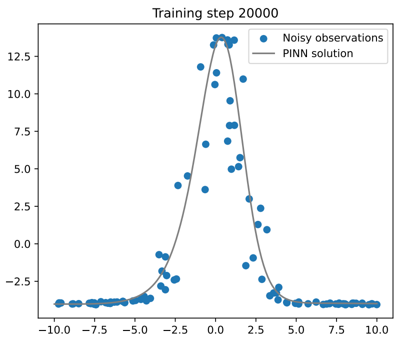
Table 1: Estimated values of parameters with analytical solution of Eq. (2)

MLE			
Parameters	Initialized	Exact	Estimated
α	5	1	0.3664
β	3	4	-2.001
γ	4	1	0.2213
PINN			
Parameters	Initialized	Exact	Estimated
α	5	1	1.0229
β	3	4	3.9868
γ	4	1	1.0105

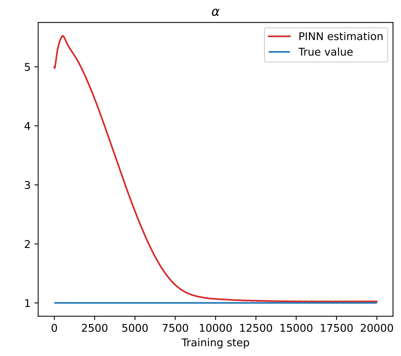
Table 2: Estimated values of parameters with analytical solution of Eq. (3)

MLE			
Parameters	Initialized	Exact	Estimated
α	5	2	0.6037
β	2.5	0	0.1417
γ	4	1	0.1802
PINN			
Parameters	Initialized	Exact	Estimated
α	5	2	1.9998
β	2.5	0	0.03891
γ	4	1	1.0244

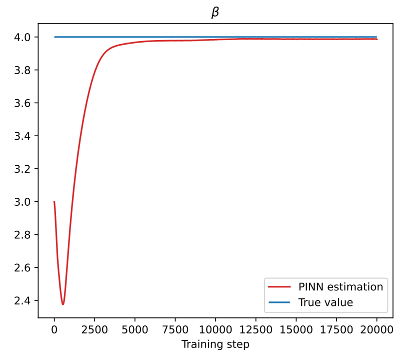
As demonstrated in Figures 2 and 3, after 20,000 training steps, the solutions produced by the PINN closely match the noisy data generated by the analytical solutions for both Eqs. (2) and (3). Then for each parameter of (2) and (3) we plotted the estimated values during learning of the network. The estimated parameter values converge to their true values.



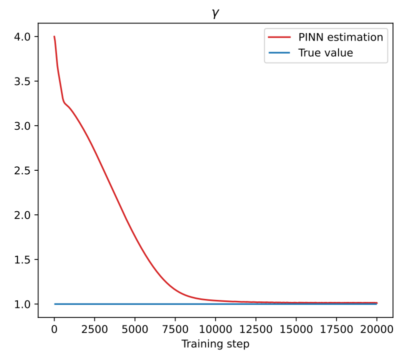
(a)



(b)

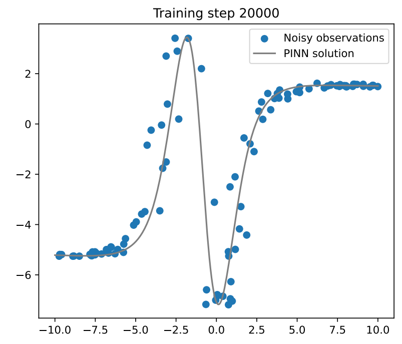


(c)

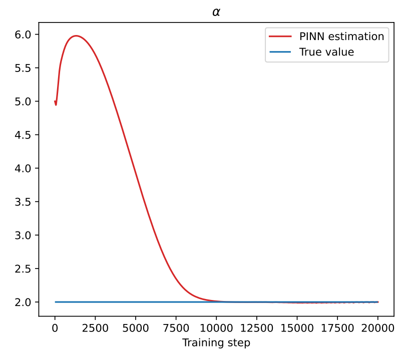


(d)

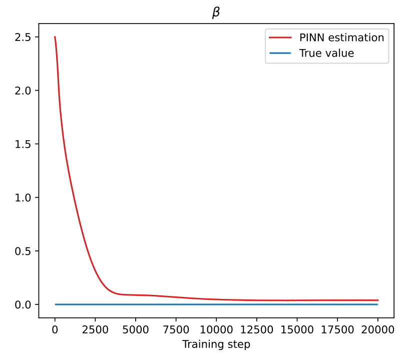
Figure 2: The PINN solution and estimated parameters for Eq. (2).



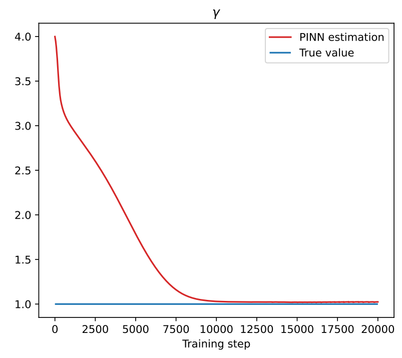
(a)



(b)



(c)



(d)

Figure 3: The PINN solution and estimated parameters for Eq. (3).

4 Conclusion

In this study, we began by introducing the Generalized Kuramoto–Sivashinsky equation and subsequently presented analytical solutions for specific parameter sets. We then estimated the parameters of this equation using two methods: MLE and PINNs, employing noisy data for our analysis. The results demonstrated that the values obtained through the PINN method exhibited high accuracy and converged closely to the exact values after an adequate number of training steps, whereas the MLE approach did not yield satisfactory results. The PINN method effectively incorporates the physical conditions of the problem during the parameter estimation process, especially when minimizing the loss function. This incorporation leads to significantly improved accuracy compared to traditional statistical methods. Furthermore, with the enhancements proposed for the PINN approach, better results can be achieved with fewer iterations, thereby increasing the accuracy of parameter estimation. Additionally, this methodology is applicable to equations that lack analytical solutions, utilizing numerical results or experimental data for estimation such as Burgers, Schrödinger, Allen-Cahn, and Navier-Stokes equations.

References

[1] M. Dehghan and V. Mohammadi. Two-dimensional simulation of the damped Kuramoto–Sivashinsky equation via radial basis function-generated finite difference scheme combined with an exponential time discretization. *Engineering Analysis with Boundary Elements*, 107:168–184, 2019.

[2] H. Lai and C. Ma. Lattice Boltzmann method for the generalized Kuramoto–Sivashinsky equation. *Physica A: Statistical Mechanics and its applications*, 388(8):1405–1412, 2009.

[3] M. Lakestani and M. Dehghan. Numerical solutions of the generalized Kuramoto–Sivashinsky equation using B-spline functions. *Applied Mathematical Modelling*, 36(2):605–617, 2012.

[4] T. Tatsumi. Irregularity, regularity and singularity of turbulence. *Turbulence and chaotic phenomena in fluids*, pages 1–10, 1984.

[5] Y. Kuramoto and T. Tsuzuki. Persistent propagation of concentration waves in dissipative media far from thermal equilibrium. *Progress of theoretical physics*, 55(2):356–369, 1976.

[6] R. J. Rossi. Mathematical statistics: An introduction to likelihood based inference. *John Wiley & Sons*, 2018.

[7] A. Raue, C. Kreutz, T. Maiwald, J. Bachmann, M. Schilling, U. Klingmüller, and J. Timmer. Structural and practical identifiability analysis of partially observed dynamical models by exploiting the profile likelihood. *Bioinformatics*, 25(15):1923–1929, 2009.

[8] D. Wu and S. Kim. Problems in the estimation of the key parameters using MLE in lung cancer screening. *Journal of clinical research and reports*, 5(3), 2020.

[9] S. Wang, S. Sankaran, H. Wang, and P. Perdikaris. An expert's guide to training physics-informed neural networks. *arXiv preprint arXiv:2308.08468*, 2023.

[10] I. E. Lagaris, A. Likas, and D. I. Fotiadis. Artificial neural networks for solving ordinary and partial differential equations. *IEEE Transactions on Neural Networks*, 9(5):987–1000, 1998.

[11] M. Raissi, P. Perdikaris, and G. 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] N. Ketkar, J. Moolayil, N. Ketkar, and J. Moolayil. Deep Learning with Python: Learn Best Practices of Deep Learning Models with PyTorch. *Apress LP*, 2020.

[13] A. P. Browning and M. J. Simpson. Geometric analysis enables biological insight from complex non-identifiable models using simple surrogates. *PLoS Computational Biology*, 19(1):e1010844, 2023.

[14] S. Cai, Z. Wang, S. Wang, P. Perdikaris, and G. E. Karniadakis. Physics-informed neural networks for heat transfer problems. *Journal of Heat Transfer*, 143(6):060801, 2021.

[15] A. G. Baydin, B. A. Pearlmutter, A. A. Radul, and J. M. Siskind. Automatic differentiation in machine learning: a survey. *Journal of Machine Learning Research*, 18(153):1–43, 2018.

bibitembaydin2018automatic A. G. Baydin, B. A. Pearlmutter, A. A. Radul, and J. M. Siskind. Automatic differentiation in machine learning: a survey. *Journal of Machine Learning Research*, 18(153):1–43, 2018.