Bridging PINNs and KANs to Handle Noisy Partial Differential Equations

Abstract

Physics-Informed Neural Networks (PINNs) are deep learning models that embed the governing physics of a system into their loss function, enabling them to solve forward and inverse Partial Differential Equations (PDEs) efficiently. However, PINNs struggle with noisy input data, which limits their effectiveness in solving PDEs in real-world applications. In this work, we propose an integrated framework that leverages Kolmogorov-Arnold Networks (KANs) to denoise the input data before passing it to PINNs for solving forward PDEs. We then demonstrate the improvement in terms of robustness, scalability, and accuracy, as evaluated on the Burgers', Heat, and Wave equations.

1 INTRODUCTION

Partial differential equations (PDEs) have been proven to be central to the modelling and analysis of intricate physical systems across a wide range of disciplines like fluid dynamics [Elman et al., 2014] electromagnetism [Taylor, 1996], and wave propagation [Davis, 2012]. Traditional numerical solvers like finite difference [Smith [1978]] and finite element [an Ying, 2007] methods have been used to numerically approximate PDE solutions. Although these solvers work well in ideal situations, they tend to fail when noise is present [Bajaj et al., 2021] [Pilar and Wahlstrom, 2022]. Real-world situations-marked by measurement errors, environmental variability, and computational imprecision, pose challenges that traditional solvers are not well-suited to address.

Machine learning techniques have been shown to be effective at solving PDEs [Chan-Wai-Nam et al., 2019] [Huré et al., 2019] [Mishra, 2018] [Liang et al., 2021]. Of these machine learning techniques, Physics-Informed Neural Networks (PINNs) have also been shown to be a very promising alternative for solving PDEs [Cai et al., 2021] [Yang et al., 2021] [Kharazmi et al., 2019] [Pang et al., 2018]. PINNs incorporate the governing equations into their loss functions, and they are capable of approximating solutions even with sparse data and in complicated geometries [Raissi et al., 2019]. PINNs have critical drawbacks when there is noise in the data or in boundary conditions. Experiments have shown that noisy inputs result in poor convergence, incorrect solutions, and a loss of the network's capacity to generalize.

1.1 RELATED WORK

Despite the significant advancements made in the field of PINNs, their sensitivity to noise remains a notable challenge. Several previous studies have explored novel techniques to enhance their robustness. For instance, Bajaj et al. [2023] had proposed Gaussian process (GP) smoothing in order to mitigate measurement noise and provide uncertainty quantification through GP variance or boundary data. However, this approach is limited to Gaussian noise and does not account for non-Gaussian noise which is more prevalently encountered in real-world scenarios. To address this limitation, Pilar and Wahlstrom [2022] proposes the PINN-EBM that uses an Energy-Based Model to learn the homogenous noise distribution in order to tackel the problem. Although the framework is efficient against nonzero mean and non-Gaussian noise, the computational overhead remains a significant drawback.

1.2 PARTIAL DIFFERENTIAL EQUATIONS IN FOCUS

To explore and address the challenges of solving noisy PDEs, we focus on three representative equations: the Burgers' equation, the Heat equation, and the Wave equation.

1.2.1 Burgers' Equation

Burgers' equation is a nonlinear PDE commonly used to model shock waves, turbulence, and other nonlinear wave phenomena. It combines advective and diffusive effects, making it a benchmark for testing numerical methods and machine learning models. The equation is expressed as:

$$\nu\frac{\partial^2 u}{\partial x^2} = \frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x}$$

where:

- $u(x,t)$ is the velocity as a function of space x and time t.
- $\frac{\partial u}{\partial t}$ represents the difference in velocity with respect to time.
- $u\frac{\partial u}{\partial x}$ is the nonlinear advection term, which is responsible for wave steepening and potential shock formation.
- $\nu\frac{\partial^2 u}{\partial x^2}$ is the diffusion term, where ν controls the smoothing effect due to diffusion.

The interaction between nonlinearity and diffusion makes Burgers' equation particularly sensitive to noise. Even minor perturbations in initial or boundary conditions can lead to irregular shock formations, chaotic flow structures, or enhanced turbulence-like behaviour.

1.2.2 Heat Equation

The Heat Equation is an elemental PDE that describes how heat in a given medium diffuses over time. It is widely applied in the fields of physics and engineering to model heat conduction and diffusion processes. It is expressed as:

$$\frac{\partial u}{\partial t} = \alpha\frac{\partial^2 u}{\partial x^2}$$

where:

- $u(x,t)$ represents the temperature or concentration of a diffusing quantity at position x and at time t.
- $\frac{\partial u}{\partial t}$ specifies how the temperature changes over time.
- $\alpha\frac{\partial^2 u}{\partial x^2}$ is the diffusion term, where α represents thermal diffusivity and controls how quickly the heat spreads through the given medium.

Since heat diffusion is a smoothing process, any small perturbations in the initial or boundary conditions tend to fade over time. However, stochastic fluctuation introduces unpredictable temperature variations, while errors in numerical calculations make the solution unstable.

1.2.3 Wave Equation

The Wave Equation describes how waves, such as sound waves, water waves, and electromagnetic waves, propagate through a medium. This makes the equation a fundamental in fields of physics and engineering for understanding vibrations and oscillations. It is given by:

$$\frac{\partial^2 u}{\partial t^2} = c^2\frac{\partial^2 u}{\partial x^2}$$

where:

- $u(x,t)$ represents the wave function.
- $\frac{\partial^2 u}{\partial t^2}$ represents the wave's acceleration over time.
- $c^2\frac{\partial^2 u}{\partial x^2}$ represents the curve's spatial curvature, where c is the wave speed.

Unlike diffusion processes, waves preserve disturbances, making the Wave Equation highly sensitive to noise in initial or boundary conditions leading to even small perturbations causing lasting distortions in wave behaviour. However, numerical errors can distort the wave or change its speed.

1.3 SKEWED NORMAL NOISE

Skewed Normal Noise is a type of probabilistic distribution that extends the conventional normal (Gaussian) distribution by introducing asymmetry, i.e., making it skewed to the right or left, therefore making it a more flexible model for noise patterns observed in real-world, where deviations from symmetry are common. The probability density function (PDF) of a skewed normal distribution is as follows:

$$f(x) = \frac{2}{\omega\sqrt{2\pi}}e^{-\frac{(x-\xi)^2}{2\omega^2}}\,\Phi\left(\lambda\frac{x-\xi}{\omega}\right)$$

where:

- x represents the random variable that follows a skewed normal distribution.
- ξ represents the location parameter that shifts the distribution's center.
- ω represents the scale parameter that controls the spread of the distribution.
- λ represents the skewness parameter that determines the asymmetry of the distribution.
- $\Phi(\cdot)$ is the cumulative distribution function (CDF) of the standard normal distribution.

Standard Gaussian noise assumes symmetric errors, which does not always hold in real-world scenarios. Incorporating skewed normal noise in training data ensures that the model can learn to generalize better under realistic situations.

1.4 PHYSICS-INFORMED NEURAL NETWORKS

Physics Informed Neural Networks (PINNs), also known as Theory-Trained Neural Networks (TTNs), are a class of deep feed-forward neural networks that are designed to embed physical laws that govern a particular dataset into the learning process [Raissi et al., 2019]. PINNs have been found to be extremely effective for problems that are defined by partial differential equations (PDEs), as they can integrate relevant physical principles right into their structure during the training process.

1.4.1 Architecture of a PINN

A typical PINN consists of an input layer that takes in parameters such as space (x) and time (t) with respect to a particular PDE due to the nature of PINNs; they learn a continuous function u (x,t) that can be queried at any point in time and space.

The hidden layers of a PINN are fully connected feedforward layers where the number of layers usually ranges between 5-10 layers with 50-500 neurons per layer, depending on the complexity of the problem. Additionally, the activation function used is tanh, as it assists approximate smooth physical functions (such as solutions to PDEs).

The output layer of the PINN can have one or more neurons, based on what the required prediction is.

1.4.2 Loss Function of a PINN

The loss function of a Physics-Informed Neural Network (PINN) is a key factor that distinguishes it from other neural networks and consists of three major components:

- **Data Loss**: It is computed as the error between the true values and the network's predictions:

  $$L_{\text{data}} = \sum_i \left(u_\theta(x_i, t_i) - u_{\text{true}}(x_i, t_i)\right)^2$$

  Data loss ensures that the predictions made by the PINN do not deviate significantly from the known observations.

- **Physics Loss**: Since PINNs embed physics laws, they enforce them as a soft constraint in the loss function, making PINNs extremely unique.

  For example, for a PDE of the form:

  $$N[u(x,t)] = 0$$

  where N is a differential operator, it is crucial that the neural network learns a function $u_\theta(x,t)$ that satisfies the equation.

  Therefore, the physics loss of a PINN is defined as:

  $$L_{\text{physics}} = \sum_i |N[u_\theta(x_i, t_i)]|^2$$

  Physics loss ensures that the network respects the underlying physical laws at all times, even at points where data is inexistent.

- **Initial and Boundary Conditions Loss**: The network must also satisfy the initial and boundary conditions required by the PDEs.

  $$L_{\text{BC}} = \sum_i \left(u_\theta(x_i, t_i) - u_{\text{BC}}(x_i, t_i)\right)^2$$

  This loss ensures that the solutions comply with the boundary conditions by penalizing the network whenever it violates them.

Therefore, the total loss function is:

$$L(\theta) = L_{\text{data}} + L_{\text{physics}} + L_{\text{BC}}$$

1.4.3 Effect of Noise on PINNs

Physics-Informed Neural Networks (PINNs) are built and trained to satisfy two conflicting objectives:

- Replicate real-world (potentially noisy) data using L_{data}.
- Comply with the driving physics principles using L_{physics}.

However, real-world measurements are often noisy, and the problems that a PINN undergoes due to noise are as follows:

- PINNs try to minimize the Mean Squared Error (MSE) in the data loss, which leads to the PINN blindly trying to match every noisy data point. This results in overfitting and unphysical solutions.
- PINNs are trained on governing physics laws for a particular PDE, but in the presence of noisy data, the potential of a PINN getting stuck in an optimization trap is extremely high. Noisy data suggests one behavior, whereas the physics laws suggest another.
- PINNs are excellent tools for solving inverse PDEs to infer unknown parameters due to their unique nature. However, noisy data makes such PINNs highly unstable, leading to incorrect physical parameters being predicted.
- Since Automatic Differentiation (AD) is used by PINNs to compute derivatives, noisy data can lead to amplified errors in higher-order derivatives.

1.5 KOLMOGOROV-ARNOLD NETWORKS AND ROLE IN DENOISING

Kolmogorov-Arnold Networks (KANs), introduced by Liu et al. [2024], are a novel type of neural networks inspired by the Kolmogorov-Arnold Representation Theorem [Schmidt-Hieber, 2020]. Unlike conventional deep neural networks [Schmidhuber, 2015], they do not have fixed activation functions, i.e., they learn their own activation functions during training. This approach allows KANs to approximate highly complex functions with fewer parameters, which leads to increased interpretability and efficiency.

One of the biggest strengths of KANs lies in their ability to filter noise in order to learn meaningful function representations:

- Each node in a KAN learns its own smooth function instead of using a fixed activation. The same learned functions naturally smooth out high-frequency noise, if any, thereby preventing overfitting. Therefore, instead of memorizing noise, the KAN is designed to capture only meaningful variations.
- Conventional deep neural networks, which rely on stacked layers and matrix multiplication to approximate functions which leads to a higher risk of overfitting, KANs approximate functions directly with learned univariate functions. Since fewer parameters are being used, the network generalizes better, making it more resistant to noise.
- The usage of fixed activations introduces sharp changes in function approximation. A small noisy perturbation could cause a large shift in output, which is unstable and undesirable. KANs reduce such sensitivity to noise and outliers using B-spline-based activations that enforce smoother transitions between function values. KANs adjust their learned activation functions to fit general trends instead of introducing abrupt changes.

2 METHODOLOGY

2.1 PROPOSED SOLUTION

To improve the robustness of PINNs in solving forward PDEs under noisy conditions, we propose a hybrid framework that integrates KANs with PINNs. 1 In our approach, KAN acts a a preprocessing module which refines noisy inputs (initial and boundary conditions) to recover high-fidelity representations before they are fed into the PINN. The PINN then leverages these denoised inputs to solve the forward PDE. This synergy allows us to address the inherent noise sensitivity of PINNs, enabling robust and accurate predictions even under challenging conditions.

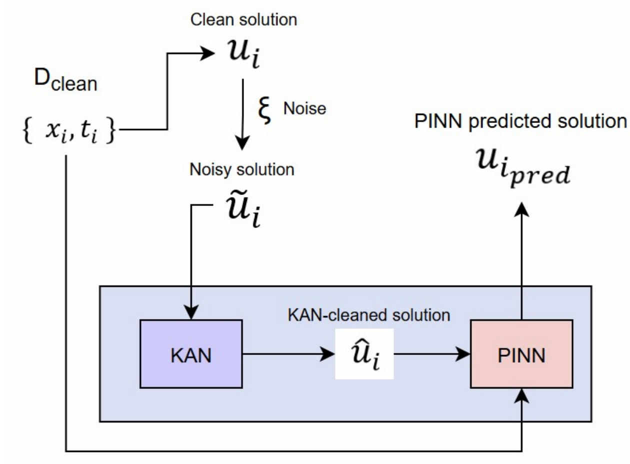

Figure 1: KAN-PINN framework for solving forward PDEs by denoising noisy inputs with KAN before predicting the solution using PINN.

2.2 DATA

The spatial and temporal domains were generated through appropriate discretization to ensure numerical stability, adhering to the Courant-Friedrichs-Lewy [Weisstein, 2002] (CFL) condition wherever applicable.

The clean solutions for each PDE were generated using different numerical methods.

1. The Heat equation was solved using the Runge-Kutta method [Butcher, 2007] from SciPy's solve_ivp solver.
2. The Wave equation was solved using a finite difference approach, where Forward Euler was used for initialization, and updates were performed based on the wave equation scheme, with initial conditions consisting of a Gaussian pulse displacement, zero velocity, and a random scaling factor.
3. The Burgers' equation was computed using a finite difference scheme. [Takamoto et al., 2024]

The noisy solutions are generated by adding the skewed normal noise to the clean solutions as follows:

$$u_{\text{noisy}} = u_{\text{clean}} + noise$$

The resulting datasets-clean and noisy solutions along with the spatial and temporal domains- were stored in an HDF5 file for further processing.

2.3 DENOISING USING KAN

The set of noisy solutions generated is passed as input to the KAN, which is trained to denoise and reconstruct solutions of the PDE from noisy inputs.

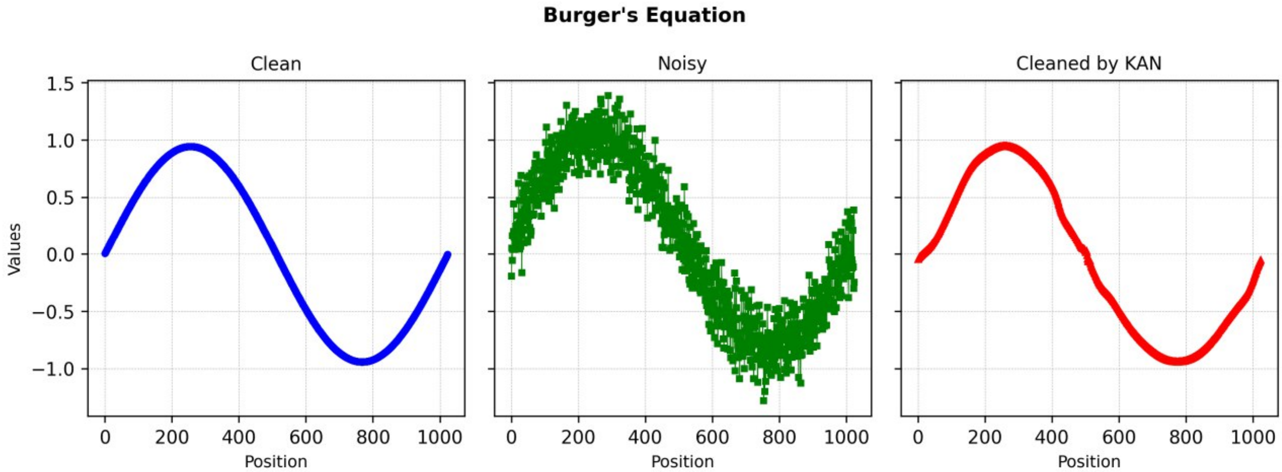

Figure 2: Reconstruction of Burgers' Equation using KAN at t=0. The clean data (left) represents the ground truth, while the noisy data (middle) contains artificially introduced perturbations. The cleaned output (right) is obtained after denoising using a Kolmogorov-Arnold Network (KAN).

The implemented KAN architecture consists of multiple KANLinear layers that replace conventional fully connected layers with adaptive spline-based transformations.

The forward pass of a KANLinear layer is given by the weighted sum of the learnable activation and the B-spline basis functions [Piegl et al., 1995].

The set of noisy solutions is divided into mini batches and trained for a finite number of epochs. This is supplemented by the splitting and shuffling of the time stamps into distinct time steps in order to ensure that the KAN is able to independently learn how the solution is supposed to look at each time step and therefore remove the noise.

At each iteration, the forward pass of KAN computes the predicted outputs after which the MSE loss is calculated and the parameters are adjusted via backpropagation in order to minimize the MSE.

The final output obtained from the KAN model is a denoised reconstruction of the PDE's solutions 2. This output is an approximate cleaned version of the input data, effectively filtering out any noise that is present while preserving the underlying dynamics.

2.4 PREDICTING SOLUTIONS USING PINN

The proposed PINN architecture consists of a fully connected feed-forward neural network with 20 alternating layers of linear transformations and hyperbolic tangent activation functions. The input to this network is a pair $(X, \hat{u})$, where $X = (x, t)$ represents the spatial and temporal coordinates, and $\hat{u}$ is the denoised solution obtained from the KAN. These inputs are obtained post the data generation and denoising stages.

The output of the network is $u_\theta(x, t)$, an approximation of $u(x, t)$, the solution of the particular PDE 3. The final layer maps the hidden representations to a single output neuron that corresponds to the predicted solution of the PDE.

A key aspect of this model is that the physical parameters are known and fixed during training.

The training involves the usage of a gradient-based optimization algorithm with a predefined learning rate. During training, the optimizer iteratively minimizes the total loss function over several epochs. The network weights are optimized to ensure accurate PDE solutions.

The trained model is evaluated on an independent test set in order to assess its ability to generalize and precisely satisfy the PDE constraints.

3 OBSERVATIONS AND RESULTS

3.1 EFFECT OF NOISE ON PINN PERFORMANCE

The introduction of noise significantly increased both the MSE and the loss across all three PDEs compared to the original clean data. From Table 1, we observe that for the Burgers' equation, the MSE rises from 0.016251 in the clean case to 0.047874 when noise is introduced, and the loss increases from 0.018702 to 0.051007. Similarly, the Heat equation exhibits a drastic rise in error, with MSE increasing by an order of magnitude from 0.004926 (clean) to 0.067647 (noisy), while the loss escalates from 0.005324 to 0.125456. The Wave equation follows the same trend, with the MSE and loss both rising considerably in the presence of noise.

These results highlight the fundamental limitation of standard PINNs when dealing with real-world data, where noise

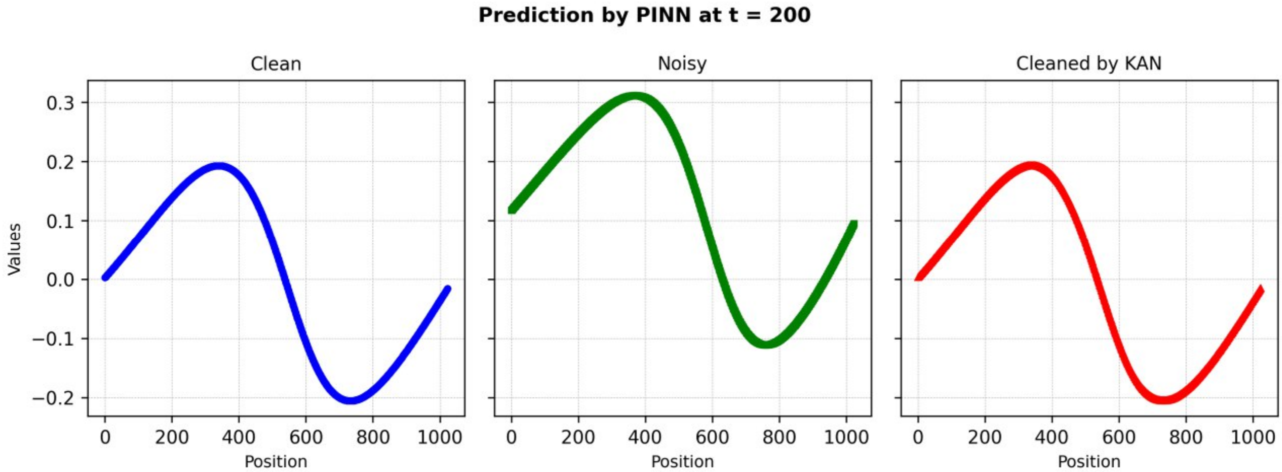

Figure 3: Prediction by PINN at t=200. The leftmost plot shows the PINN-predicted output on clean data, the middle plot displays the PINN-predicted output on noisy input data, and the rightmost plot presents the PINN-predicted output after preprocessing with KAN.

Table 1: MSE and Loss values for different PDEs under various conditions.

PDE	MSE			LOSS		
	Clean	Noisy	Cleaned by KAN	Clean	Noisy	Cleaned by KAN
Burgers	0.016251	0.047874	0.016759	0.018702	0.051007	0.019722
Heat	0.004926	0.067647	0.016839	0.005324	0.125456	0.016874
Wave	0.012826	0.040183	0.012796	0.012832	0.039806	0.012738

is inevitable. The substantial increase in error across all PDEs suggests that standard PINNs alone may not be well-suited for handling noisy data and require additional preprocessing steps to mitigate these effects.

3.2 EFFICACY OF KAN IN NOISE REDUCTION

As shown in Table 1, the application of KAN significantly reduces errors compared to the noisy case. For the Burgers' equation, the MSE after KAN processing drops from 0.047874 (noisy) to 0.016759, closely matching the clean case (0.016251). Similarly, the loss improves from 0.051007 (noisy) to 0.019722, demonstrating the effectiveness of KAN in noise mitigation.

A similar trend is observed in the Heat and Wave equations. In the case of the Heat equation, which was most affected by noise, the KAN reduces the MSE from 0.067647 (noisy) to 0.016839, bringing it closer to the clean baseline of 0.004926. The loss follows the same pattern, dropping from 0.125456 to 0.016874. For the Wave equation, the improvements are also evident, with MSE reducing from 0.040183 (noisy) to 0.012796 and loss dropping from 0.039806 to 0.012738.

These results indicate that KAN is a good denoising approach, enabling PINNs to recover from degradation in performance due to noisy input data and effectively restore predictive accuracy for PINNs on different PDEs by mapping noisy input data to a cleaner representation before feeding it into the PINN.

3.3 RESIDUAL ERROR AFTER KAN PROCESSING

While KAN heavily counteracts the harmful influence of noise, it is not able to entirely bring the performance back to its baseline in the clean data for all experiments. For instance, Table 1 indicates that in terms of MSE as well as loss, a small gap persists between "Clean" and "Cleaned by KAN.". This discrepancy is particularly noticeable in the Heat equation, where the post-KAN MSE (0.016839) remains higher than the clean baseline (0.004926), and the loss (0.016874) does not fully match the clean case (0.005324).

This implies that though KAN can filter a significant amount of noise, the PDE may be more noise-sensitive and thereby full recovery unattainable. One such reason for such variability is possibly due to the interaction between noise and structure for different equations, such as strong diffusion properties shown by the Heat equation, as opposed to Burgers and Wave equations.

Further improvement in KAN's architecture might be through further enhanced noise-adaptive training strategies, aiming at higher abilities in restoring the input for more varied PDEs. More alternatives for the denoising strategy could include hybrid KAN-PINN training in a further pursuit to bridge the gap in residual errors and eventually produce near-clean performances even when a large amount of noise is presented.

4 FUTURE WORK

Although our proposed framework demonstrates promising results in handling Skewed Normal Noise for forward PDE problems, there is ample scope for future exploration.

While our research only focuses on a specific type of noise distribution, real-world scenarios often involve more diverse and unpredictable noise patterns. One direction of future work could include testing the framework against other kinds of noises in order to assess its performance under miscellaneous conditions.

Since the current implementation centers around relatively simpler equations, a valuable advancement in this research would be to extend the framework to handle higher-order and more complex PDEs.

The dependency of KAN on the specific range of spatial and temporal variables during training remains a key limitation of our approach. Another potential direction for future work could be to redesign the KAN component in order for it to generalize across different input ranges without the need for retraining. This would notably improve the scalability of our framework.

Addressing the aspects mentioned above will help further refine the adaptability, scalability, and effectiveness of our KAN-PINN framework for a much wider range of applications.

5 CONCLUSION

The findings of this research demonstrate the effectiveness of the KAN as a preprocessing step for improving the robustness of PINNs against non-Gaussian noise. By filtering out noise from the input data, KAN significantly reduces both the MSE and the loss across all three PDEs, bringing them closer to the clean data baseline. This highlights KAN's potential in mitigating the adverse effects of noise, thereby enhancing the reliability of PINNs in solving PDE problems.

However, while KAN improves performance, its effectiveness varies across PDEs. For instance, the results from the Heat equation reveal that although the KAN-processed data is notably cleaner compared to the raw noisy input, it does not completely restore PINN performance to that of the clean data case. This proposes that certain PDEs may exhibit structural dependencies that limit the extent to which KAN can denoise any induced distortions.

Overall, the study highlights the viability of KAN as a noise reduction strategy for PINNs while also identifying potential areas for future improvements to ensure broader applicability across diverse PDEs.

6 BACK MATTER

References

Lung an Ying. Partial differential equations and the finite element method. *Math. Comput.*, 76:1693–1694, 2007. URL `https://api.semanticscholar.org/CorpusID:36929582`.

Chandrajit Bajaj, Luke McLennan, Timothy Andeen, and Avik Roy. Recipes for when physics fails: recovering robust learning of physics informed neural networks. *Machine learning: science and technology*, 4(1):015013, 2023.

Chandrajit L. Bajaj, Luke McLennan, Timothy R. Andeen, and Avik Roy. Robust learning of physics informed neural networks. *ArXiv*, abs/2110.13330, 2021. URL `https://api.semanticscholar.org/CorpusID:239885660`.

John Butcher. Runge-kutta methods. *Scholarpedia*, 2(9): 3147, 2007.

Shengze Cai, Zhicheng Wang, Sifan Wang, Paris Perdikaris, and George Em Karniadakis. Physics-informed neural networks for heat transfer problems. *Journal of Heat Transfer*, 2021. URL `https://api.semanticscholar.org/CorpusID:234819904`.

Quentin Chan-Wai-Nam, Joseph Mikael, and Xavier Warin. Machine learning for semi linear pdes. *Journal of scientific computing*, 79(3):1667–1712, 2019.

Julian L Davis. *Wave propagation in solids and fluids*. Springer Science & Business Media, 2012.

Howard C Elman, David J Silvester, and Andrew J Wathen. *Finite elements and fast iterative solvers: with applications in incompressible fluid dynamics*. Oxford university press, 2014.

Côme Huré, Huyên Pham, and Xavier Warin. Some machine learning schemes for high-dimensional nonlinear pdes. *ArXiv*, abs/1902.01599, 2019. URL `https://api.semanticscholar.org/CorpusID:59604454`.

Ehsan Kharazmi, Z. Zhang, and George Em Karniadakis. Variational physics-informed neural networks for solving partial differential equations. *ArXiv*, abs/1912.00873, 2019. URL `https://api.semanticscholar.org/CorpusID:208527690`.

Senwei Liang, Shixiao Willing Jiang, John Harlim, and Haizhao Yang. Solving pdes on unknown manifolds with machine learning. *ArXiv*, abs/2106.06682, 2021. URL `https://api.semanticscholar.org/CorpusID:235422536`.

Ziming Liu, Yixuan Wang, Sachin Vaidya, Fabian Ruehle, James Halverson, Marin Soljacic, Thomas Y. Hou, and Max Tegmark. Kan: Kolmogorov-arnold networks. *ArXiv*, abs/2404.19756, 2024. URL `https://api.semanticscholar.org/CorpusID:269457619`.

Siddhartha Mishra. A machine learning framework for data driven acceleration of computations of differential equations. *ArXiv*, abs/1807.09519, 2018. URL `https://api.semanticscholar.org/CorpusID:71145824`.

Guofei Pang, Lu Lu, and George Em Karniadakis. fpinns: Fractional physics-informed neural networks. *SIAM J. Sci. Comput.*, 41:A2603–A2626, 2018. URL `https://api.semanticscholar.org/CorpusID:119202548`.

Les Piegl, Wayne Tiller, Les Piegl, and Wayne Tiller. B-spline basis functions. *The NURBS Book*, pages 47–79, 1995.

Philipp Pilar and Niklas Wahlstrom. Physics-informed neural networks with unknown measurement noise. *ArXiv*, abs/2211.15498, 2022. URL `https://api.semanticscholar.org/CorpusID:254044055`.

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. ISSN 0021-9991. doi: https://doi.org/10.1016/j.jcp.2018.10.045. URL `https://www.sciencedirect.com/science/article/pii/S0021999118307125`.

Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural networks*, 61:85–117, 2015.

Johannes Schmidt-Hieber. The kolmogorov-arnold representation theorem revisited. *Neural networks : the official journal of the International Neural Network Society*, 137:119–126, 2020. URL `https://api.semanticscholar.org/CorpusID:220919673`.

George Davey Smith. Numerical solution of partial differential equations: Finite difference methods. 1978. URL `https://api.semanticscholar.org/CorpusID:265725981`.

Makoto Takamoto, Timothy Praditia, Raphael Leiteritz, Dan MacKinlay, Francesco Alesiani, Dirk Pflüger, and Mathias Niepert. Pdebench: An extensive benchmark for scientific machine learning, 2024. URL `https://arxiv.org/abs/2210.07182`.

Michael Taylor. Partial differential equations i: Basic theory. 1996. URL `https://api.semanticscholar.org/CorpusID:64477424`.

Eric W. Weisstein. Courant-friedrichs-lewy condition. 2002. URL `https://api.semanticscholar.org/CorpusID:133076423`.

Liu Yang, Xuhui Meng, and George Em Karniadakis. B-pinns: Bayesian physics-informed neural networks for forward and inverse pde problems with noisy data. *Journal of Computational Physics*, 425:109913, 2021.