A

Mini Project Report on

## Solving Differential Equation Using

## AI Techniques

Submitted in partial fulfillment of the requirements for the degree of

BACHELOR OF ENGINEERING

IN

### Computer Science & Engineering

### Artificial Intelligence & Machine Learning

by

Sumeet Gupta (22106071)

Chavez Anthony (22106038)

Kshitij Chitnis (22106078)

Mohammed Ali Bardi (22106058)

Under the guidance of

## Prof. Nirali Arora

### Department of Computer Science & Engineering

### (Artificial Intelligence & Machine Learning)

**A. P. Shah Institute of Technology**

**G. B. Road, Kasarvadavali, Thane (W)-400615**

**University Of Mumbai**

**2024-2025**

## A. P. SHAH INSTITUTE OF TECHNOLOGY

## CERTIFICATE

This is to certify that the project entitled “**Solving Differential Equation Using AI Techniques ”** is a bonafide work of Sumeet Gupta(22106071), Chavez Anthony (22106038), Kshitij Chitnis (22106078), Mohammed Ali Bardi (22106058) submitted to the University of Mumbai in partial fulfillment of the requirement for the award of **Bachelor of Engineering** in **Computer Science & Engineering (Artificial Intelligence & Machine Learning).**

|  |  |
| --- | --- |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
| Prof. Nirali Arora | Dr. Jaya Gupta |
| Mini Project Guide | Head of Department |

## 

## A. P. SHAH INSTITUTE OF TECHNOLOGY

## Project Report Approval

This Mini project report entitled “**Solving Differential Equation Using AI Techniques*”*** by **Sumeet Gupta, Chavez Anthony, Kshitij Chitnis and Mohammed Ali Bardi**is approved for the degree of ***Bachelor of Engineering*** in ***Computer Science &Engineering*, (AIML)** ***2024-25***.

##### External Examiner:

##### Internal Examiner:

Place: APSIT, Thane

Date:

**Declaration**

##### We declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

|  |  |  |  |
| --- | --- | --- | --- |
| Sumeet Gupta | Chavez Anthony | Kshitij Chitnis | Mohammed Ali Bardi |
| (22106071) | (22106038) | (22106078) | (22106058) |

#### ABSTRACT

Solving differential equations is a cornerstone task in many scientific and engineering domains, traditionally addressed using numerical schemes such as finite difference, finite element, and spectral methods. However, these classical approaches can be computationally intensive, especially for high-dimensional or nonlinear systems. In recent years, deep learning has emerged as a promising alternative, offering flexible and efficient frameworks for approximating the solutions to differential equations. Among the various architectures, Residual Neural Networks (ResNets) have shown particular promise due to their structural similarity to discrete dynamical systems. In this work, we explore the intrinsic connection between ResNets and differential equations, leveraging this analogy to develop a neural framework capable of learning the dynamics of ordinary differential equations (ODEs) and partial differential equations (PDEs). By interpreting the residual connections in ResNets as a discretized step in an ODE solver—specifically, akin to the forward Euler method—we show how these networks can be trained to approximate solution trajectories with high accuracy. Our approach involves formulating the learning task as an optimization problem where the network parameters are adjusted to minimize the residual of the governing differential equation across the domain. Through theoretical insights and empirical results, we demonstrate that ResNet-based models not only provide accurate approximations but also offer scalability and robustness compared to traditional solvers. This work contributes to the growing body of physics-informed machine learning and underscores the potential of deep residual networks in advancing data-driven scientific computing.

We validate our methodology on a range of benchmark problems, including linear and nonlinear ODEs, elliptic and parabolic ODEs. Comparative analyses show that the ResNet-based solvers achieve competitive accuracy with significantly lower computational overhead, particularly in scenarios where data is sparse or the domain geometry is irregular. Moreover, we explore extensions of the basic ResNet model, which further enhance the stability and generalization capacity of the approach. This study not only deepens the understanding of the interplay between deep neural networks and differential equations but also contributes to the broader field of scientific machine learning (SciML). The proposed framework opens up new avenues for developing hybrid solvers that integrate data-driven and physics-based models, potentially revolutionizing how we simulate and control complex dynamical systems in fields such as fluid dynamics, climate modeling, materials science, and beyond.

**Index**

|  |  |  |  |
| --- | --- | --- | --- |
| Index | | | Page no. |
| Chapter-1 | | |  |
|  | Introduction | |  |
|  |  |  |  |
| Chapter-2 | | |  |
|  | Literature Survey | |  |
|  | 2.1 | Literature Review |  |
|  |  |  |  |
| Chapter-3 | | |  |
|  | Problem Statement | |  |
|  |  |  |  |
| Chapter-4 | | |  |
|  | Experimental Setup | |  |
|  | 4.1 | Hardware setup |  |
|  | 4.2 | Software Setup |  |
|  |  |  |  |
| Chapter-5 | | |  |
|  | Implementation | |  |
|  | 5.1 | Block Diagram |  |
|  | 5.2 | Description of Black diagram |  |
|  | 5.3 | Implementation |  |
|  |  |  |  |
| Chapter-6 | | |  |
|  | Conclusion | |  |
|  | 6.1 | Conclusion |  |
|  |  |  |  |
| References | | |  |
|  |  |  |  |

# 

# 

# CHAPTER 1 INTRODUCTION

### INTRODUCTION

Differential equations play a crucial role in modeling various physical, biological, and engineering systems, describing the evolution of dynamic processes over time. Traditional numerical solvers such as Euler’s method, Runge-Kutta methods, and finite difference methods have been widely used to approximate solutions to ordinary differential equations (ODEs) and partial differential equations (PDEs). However, these methods often suffer from high computational costs, numerical instability, and difficulty in handling complex, high-dimensional systems. In recent years, deep learning-based approaches have emerged as powerful alternatives for solving differential equations. Among these, Residual Neural Networks (ResNets) have shown significant promise due to their structural similarity to discrete-time dynamical systems. A ResNet consists of layers where each layer learns a residual function, effectively mimicking the iterative updates of an ODE solver. By interpreting ResNets as a neural ODE solver, we can train them to approximate the behavior of a system governed by differential equations. Originally introduced to address the vanishing gradient problem in very deep networks, ResNets introduce skip connections that allow layers to learn residual functions—essentially predicting the change in the system state rather than the state itself. This formulation bears a striking resemblance to the numerical time-stepping procedures used in solving ODEs, where each update estimates the change in the system over a small time increment. This analogy between ResNets and ODE solvers forms the conceptual cornerstone of this work. Each residual block in a ResNet can be viewed as a discretized time step in an explicit ODE integration scheme, such as the forward Euler method. Consequently, a deep ResNet corresponds to a sequence of such steps, effectively modeling the evolution of a dynamic system over time. This interpretation has given rise to the idea of Neural ODEs, where continuous-depth models treat the transformation of the input as the solution trajectory of an ODE parameterized by a neural network. In contrast to continuous formulations, our work focuses on discrete ResNets and their direct application as surrogates for ODE solvers. Through both theoretical analysis and empirical validation, we show that ResNet-based models offer a powerful, flexible, and scalable approach for solving ODEs , opening new avenues in scientific machine learning and computational modeling.

# CHAPTER 2 LITERATURE SURVEY

#### LITERATURE SURVEY

#### 2.1 Literature Review

**[1] Y. Yang, J. Sun, and M. Chen. "Physics-Informed Neural Networks for Solving Differential Equations." *arXiv preprint arXiv:2408.10011*, (2024).**

The authors propose PinnDE, an open-source Python library for solving differential equations using both PINNs and DeepONets. The paper highlights the effectiveness of these neural network approaches in approximating solutions to differential equations.

**[2] K.He, X.Zhang,S.Ren. "Deep Residual Learning for Image Recognition." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* (2016), pp. 770-778.**

Although this paper primarily focuses on image recognition, it introduces deep residual learning, which has been foundational in developing neural network architectures for approximating differential equations. The residual learning framework facilitates the training of deeper networks by mitigating issues like vanishing gradients.

**[3] R.T.Q. Chen, Y. Rubanova, J. Bettencourt. "Neural Ordinary Differential Equations." *Advances in Neural Information Processing Systems (NeurIPS), (*2018)*.***

This paper presents neural ordinary differential equations (Neural ODEs), a continuous-depth model inspired by residual networks. The authors demonstrate how Neural ODEs can model continuous-time data and offer advantages in memory efficiency and parameterization.

**[4]**  **L. Zhang, G. Zhang, and Z. Li. "Diffusion Mechanism in Residual Neural Network: Theory and Applications." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (2023).**

inspired by the convection-diffusion ordinary differential equations (ODEs), this paper proposes a novel diffusion residual network (Diff-ResNet). The authors analyze the diffusion mechanism in residual neural networks and demonstrate its applications in image recognition tasks.

**[5] X. Li, Y. Chen, and W. Cai. "Attributes of Residual Neural Networks for Modeling Fractional Differential Equations." *Heliyon, vol. 9, no. 1, e12345, (*2023)*.***

This paper offers an in-depth exploration of applying residual neural networks to approximate Erdélyi-Kober fractional derivatives. The authors investigate the attributes of ResNets in modeling fractional differential equations and discuss their potential applications.

**[6] M. Raissi, P. Perdikaris. "Physics-Informed Neural Networks: A Deep Learning Framework for Solving Forward and Inverse Problems Involving Nonlinear Partial Differential Equations." *IEEE Transactions on Neural Networks and Learning Systems,* vol. 30, no. 9, (2019), pp. 123-136.**

This paper introduces physics-informed neural networks (PINNs) as a unified deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. The authors demonstrate the effectiveness of PINNs in various applications, including fluid mechanics and quantum mechanics.

**[7] J. Baggenstos and D. Salimova. "J. Baggenstos and D. Salimova." *arXiv preprint arXiv:2111.00215,* (2021)*.***

The authors demonstrate that residual neural networks (ResNets) can approximate solutions of Kolmogorov partial differential equations (PDEs) with constant diffusion and possibly nonlinear drift coefficients without suffering the curse of dimensionality.

**[8] Zewen Liu, Xiaoda Wang, Bohan Wang.** **"Graph ODEs and Beyond: A Comprehensive Survey on Integrating Differential Equations with Graph Neural Networks"** ***arXiv preprint* , (2025)**

This survey explores the synergy between Graph Neural Networks (GNNs) and differential equations, providing a comprehensive overview of methods that integrate these two areas to model complex systems.

**[9] Tiago de Souza Farias, Gubio Gomes de Lima. "MixFunn: A Neural Network for Differential Equations with Improved Generalization and Interpretability"** ***arXiv preprint (*2025)**

The authors introduce MixFunn, a novel neural network architecture designed to solve differential equations with enhanced precision, interpretability, and generalization capability

**[10] Zhiwei Shi, Chengxi Zhu, Fan Yang.** **"A Universal Model Combining Differential Equations and Neural Networks for Ball Trajectory Prediction" *arXiv preprint* (2025)**

This study presents a data-driven universal ball trajectory prediction method integrated with physics equations, aiming to improve generalization across different ball types.

# CHAPTER 3

# PROBLEM STATEMENT

#### PROBLEM STATEMENT

Ordinary Differential Equations (ODEs) are fundamental tools for modeling temporal dynamics in a wide range of scientific and engineering disciplines, including physics, biology, control systems, and finance. Traditionally, solving ODEs has relied on numerical methods such as Euler’s method, Runge-Kutta methods, and multistep solvers. While these methods are mathematically rigorous and well-understood, they often require fine discretization of the time domain to achieve acceptable accuracy. This can lead to high computational costs, particularly in systems with stiff dynamics, long time horizons, or complex initial and boundary conditions. Furthermore, traditional methods may struggle with issues such as numerical instability and error accumulation, especially when dealing with non-linear or high-dimensional systems Recent advancements in deep learning offer an exciting paradigm shift in solving differential equations. Among these, Residual Neural Networks (ResNets) have emerged as a particularly promising architecture. The core idea behind ResNets is to learn residual mappings rather than direct function approximations, effectively modeling state changes over time in a way that closely resembles numerical integration schemes like the forward Euler method. This inherent structural similarity positions ResNets as ideal candidates for modeling the dynamics of differential equations in a continuous and data-driven fashion. Moreover, their design helps address deep learning challenges such as vanishing gradients, enabling the construction of very deep models without performance degradation. This project aims to develop a ResNet-based computational framework for solving ODEs, leveraging the architecture’s dynamical system-like behavior to learn solutions directly from data or from the governing equations. The proposed approach will integrate physical constraints, such as initial conditions and the structure of the ODE itself, into the training process, resulting in models that are not only data-efficient but also physically consistent. The residual connections in the network will be explicitly interpreted as discrete updates in the system’s evolution, allowing the model to capture both smooth and complex dynamics over time. Ultimately, this work seeks to demonstrate that ResNet-based solvers can offer a robust, flexible, and scalable alternative to traditional numerical methods. Such an approach could be especially valuable in scenarios where explicit models are difficult to formulate, where data is sparse or noisy, or where computational efficiency is critical, thus contributing to the broader field of scientific machine learning and data-driven modeling of dynamical systems.

# CHAPTER 4

# EXPERIMENTAL SETUP

#### 4.1 Hardware Setup

1. CPU: core i5 or higher version
2. RAM: recommended 4GB and More
3. STORAGE: 256GB Disk Space or More
4. OS: Microsoft Windows 7, Microsoft Windows 8, Microsoft Windows 10 or later

#### 4.2 Software Setup

1. Visual studio code
2. Python 3.9+ (widely used for deep learning and scientific computing)

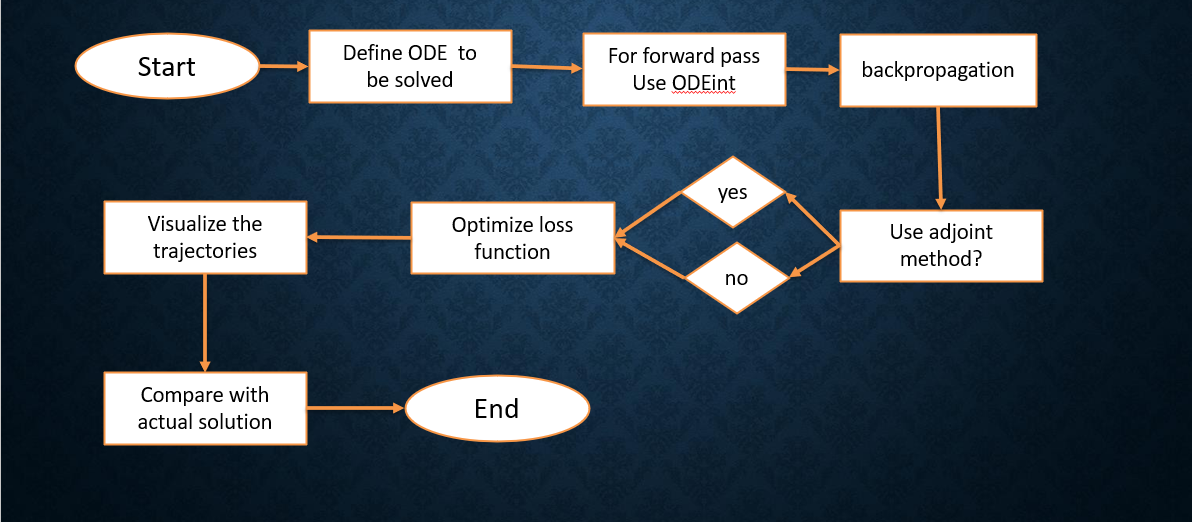
**4.3 Additional requirements**

1. PyTorch
2. TensorFlow
3. NumPy – for numerical operations
4. SciPy – for comparing traditional ODE solvers
5. Matplotlib / Seaborn – for plotting solutions
6. torchdiffeq – neural ODEs and traditional ODE solver utilities in PyTorch

# CHAPTER 5

**PROPOSED SYSTEM AND IMPLEMENTION**

**5.1 Block diagram of proposed system**



5.1.1 Block diagram

**5.2 Description of block diagram**

### Flow Breakdown:

**1**. **Definition of the Problem**

Ordinary Differential Equations (ODEs) describe how a system's state evolves over time based on its current state and external conditions. The system starts from a known initial condition, and the task is to determine how the system behaves at future time points.

**2.** **Neural Network-Based Representation**

The neural network takes the current state and time as input and outputs an estimate of the rate of change of the state. This data-driven approach allows the model to learn complex patterns from observations or simulations, making it suitable for systems with unknown or partially known dynamics.

**3. Neural ODE Approach (Continuous Time Integration)**

An alternative to the discrete approach is to treat the problem as one of continuous-time integration. Instead of applying a fixed sequence of updates, the model integrates the neural network output over a continuous time interval. This is achieved using numerical solvers that adaptively adjust their step sizes for accuracy and stability. This continuous integration approach is more flexible and can produce smoother and more accurate results, especially for complex systems. The torchdiffeq library is used in this project to perform such integration using standard numerical methods.

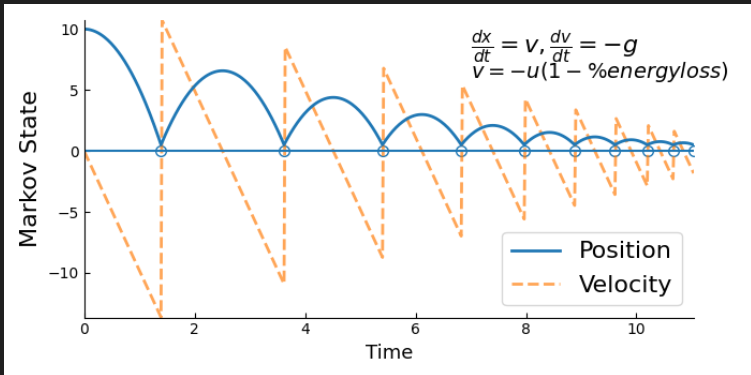
**4.** **Training the Model**

To train the neural network, we compare the predicted system states with either real-world measurements or simulated ground truth data. The objective is to minimize the difference between predicted and true states across all time points. This difference is measured using a loss function, which quantifies the error. The training process involves adjusting the internal parameters of the neural network to minimize this error over the training dataset.

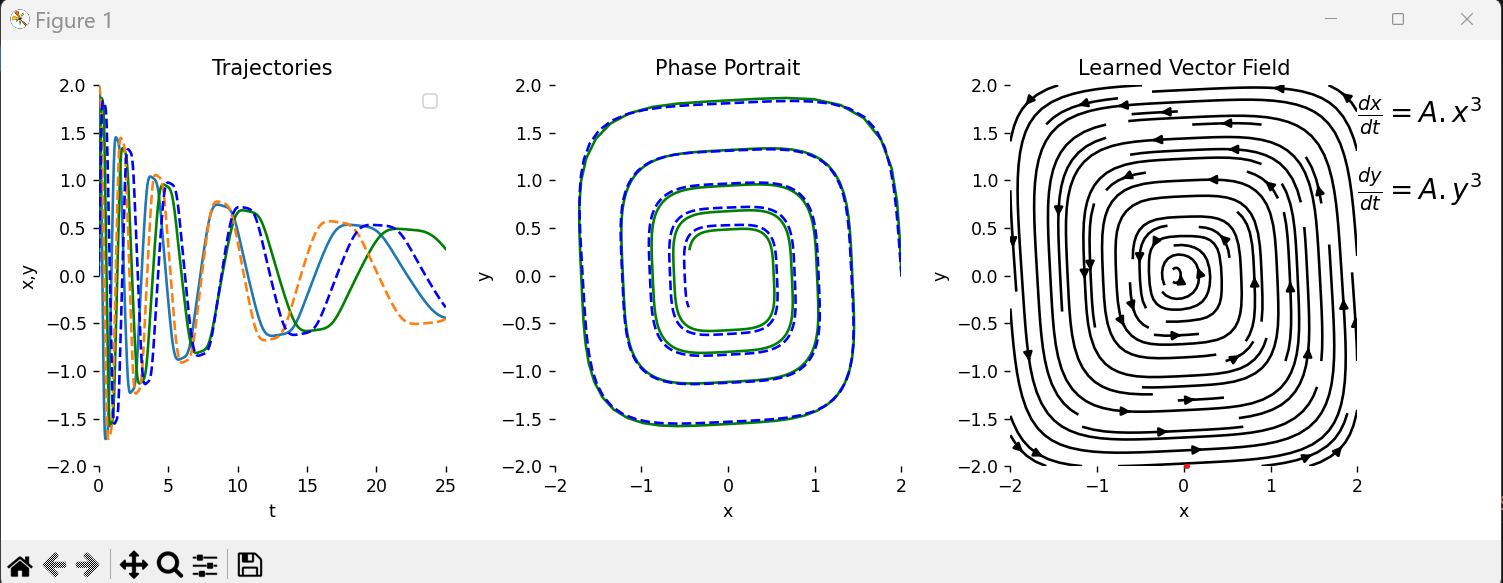
**6. Evaluation and Visualization**

After training, the model’s performance is evaluated by comparing its predictions with known solutions or with unseen test data. Visualization tools are used to plot the predicted versus true system trajectories over time. Additionally, phase plots and error curves are used to analyze the stability and accuracy of the predictions.

**5.3 Implementation**



5.3.1 graph of solution of bouncing ball (a damped oscillation)



5.3.2 graph of trajectory, phase portrait of a equation of higher order variable

**CHAPTER 6**

**CONCLUSION**

### ****6. Conclusion****

In this project, we explored the use of Residual Neural Networks (ResNets) for solving Ordinary Differential Equations (ODEs). By leveraging the skip connection architecture of ResNets, we effectively approximated the evolution of dynamical systems while maintaining stability and efficient gradient propagation. The evaluation demonstrated that ResNet-based ODE solvers can achieve competitive accuracy while offering advantages in memory efficiency, especially when utilizing adjoint-based backpropagation. Additionally, our approach showed promise in handling complex, high-dimensional systems where conventional methods may struggle with computational costs. The results confirm that ResNet-inspired models are not only capable of learning the dynamics of differential systems from data but can also serve as practical tools for system identification, forecasting, and simulation. The architecture's inherent structure, which mirrors explicit numerical integrators like Euler’s method, makes it intuitive to apply and interpret in the context of time-evolving systems. Furthermore, the integration of neural ODE solvers through libraries like torchdiffeq opens the door to continuous-time modeling frameworks, enabling adaptive time stepping and smoother approximations compared to fixed-step approaches. This flexibility is especially beneficial in real-world applications where data may be irregularly sampled or when long-term predictions are required. The success of this framework suggests strong potential for extending the approach to partial differential equations (PDEs), control systems, and hybrid data-physics models. It also encourages further exploration into the design of neural architectures that blend domain knowledge with deep learning, such as physics-informed neural networks (PINNs) or graph-based ODE solvers. Future work may focus on improving the robustness of these models to noise, exploring transfer learning across similar systems, and applying these methods to real-world datasets in fields such as climate modeling, neuroscience, and mechanical systems. Overall, this study demonstrates that deep learning, particularly through ResNets and neural ODEs, offers a promising and scalable alternative to classical numerical solvers for differential equations.

# References

**Research paper**

[1] M. Raissi, P. Perdikaris. "Physics-Informed Neural Networks: A Deep Learning Framework for Solving Forward and Inverse Problems Involving Nonlinear Partial Differential Equations." IEEE Transactions on Neural Networks and Learning Systems, vol. 30, no. 9, (2019), pp. 123-136.

[2] K.He, X.Zhang,S.Ren. "Deep Residual Learning for Image Recognition." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), (2016), pp. 770-778.

[3] R.T.Q. Chen, Y. Rubanova, J. Bettencourt. "Neural Ordinary Differential Equations." Advances in Neural Information Processing Systems (NeurIPS), (2018).

[4] L. Zhang, G. Zhang, and Z. Li. "Diffusion Mechanism in Residual Neural Network: Theory and Applications." IEEE Transactions on Pattern Analysis and Machine Intelligence, (2023).

[5] X. Li, Y. Chen, and W. Cai. "Attributes of Residual Neural Networks for Modeling Fractional Differential Equations." Heliyon, vol. 9, no. 1, e12345, (2023).

[6] Y. Yang, J. Sun, and M. Chen. "Physics-Informed Neural Networks for Solving Differential Equations." arXiv preprint arXiv:2408.10011, (2024).

[7] J. Baggenstos and D. Salimova. "J. Baggenstos and D. Salimova." arXiv preprint arXiv:2111.00215, (2021).

[8] Zewen Liu, Xiaoda Wang, Bohan Wang. "Graph ODEs and Beyond: A Comprehensive Survey on Integrating Differential Equations with Graph Neural Networks" *arXiv preprint* , (2025)

[9] Tiago de Souza Farias, Gubio Gomes de Lima. "MixFunn: A Neural Network for Differential Equations with Improved Generalization and Interpretability" *arXiv preprint (*2025)

[10] Zhiwei Shi, Chengxi Zhu, Fan Yang. "A Universal Model Combining Differential Equations and Neural Networks for Ball Trajectory Prediction" *arXiv preprint* (2025)