Physically-informed machine learning for modelling the dynamics of plant-pathogens molecular interactions

Prabal Ghosh[1][0009−0004−3449−5811]

Universite Cote d'Azur, Sophia Antipolis, France
prabal5ghosh@gmail.com

1 Research Project

I worked under the guidance of Silvia Bottini, Junior Professor Chair INRAe/UniCA, Sophia-Antipolis.

2 Abstract

Horticulture, as a critical branch of agriculture, has significantly contributed to the development of human civilization [17]. Plants, however, face ongoing challenges from biotic threats such as pathogens, which require them to employ sophisticated defense mechanisms. These defense systems involve complex signaling networks responsible for surveillance, perception, and activation of immune responses. The effectiveness of these processes is influenced by various spatial and temporal factors. The molecular dialogue between pathogens and plant hosts unfolds over time, ultimately determining the success or failure of an infection. With the advent of omics technologies, particularly transcriptomics, we now have the ability to study these intricate biological systems at a molecular level. Transcriptomics allows for the quantification of gene expression changes over time, providing valuable insights into plant responses during pathogen attacks. However, traditional methods for analyzing time-course transcriptomics data often treat each time point independently or rely on profile analysis techniques that fail to capture the temporal continuity of the data. Although alternative methods, such as regression and spline models, exist, they often fall short in providing mechanistic interpretations of the data. Furthermore, high-resolution temporal transcriptomic analysis in plant tissues is challenging due to limitations in longitudinal experiments, often involving only a few time points. This creates difficulty in drawing statistically significant conclusions regarding the changes that occur over the course of an infection.

To address these challenges, this research leverages the potential of Physics-Informed Neural Networks (PINNs), particularly inverse Physics-Informed Neural Networks (IPINN) and Neural Ordinary Differential Equations (Neural ODE) as advanced framework for analyzing time-dependent transcriptomics data. PINNs integrate observational data with underlying physical principles, making them especially suitable for handling high-dimensional, noisy, and sparse time-series datasets. Despite their success in solving various scientific problems, PINNs have yet to be fully explored in omics domains, likely due to the generally unknown governing physical systems[5] [3]. This study investigates the application of IPINNs and Neural ODEs, to analyze longitudinal multi-transcriptomics data related to plant defense responses. These methods enable the integration of data-driven approaches with physical constraints, allowing for a deeper understanding of the temporal dynamics of plant-pathogen interactions. By addressing limitations in traditional analyses, this framework aims to uncover new insights into the molecular mechanisms underlying plant defense, offering a novel perspective on pathogen-related dynamics even in the presence of noisy and incomplete data. This work represents a significant step toward advancing the use of machine learning in systems biology and enhancing our understanding of plant immune responses.

Keywords: Physics Informed Machine Learning, Physics-Informed Dynamical Variational Autoencoder, Partial Differential Equation, Ordinary Differential Equation, Stochastic Differential Equation, Extended Kalman Filter, Discretise Statistical Finite Element Method, Monte Carlo, Latent State-Space Model, Neural Ordinary Differential Equation

3 Introduction

3.1 Transcriptomics and Plant Immune Responses

Omics technologies have significantly advanced the study of biological systems by enabling detailed molecular-level analysis of complex processes. These high-throughput methods encompass genomics, transcriptomics, proteomics, and metabolomics, each offering insights into different biological aspects [16].

Transcriptomics analyzes gene expression changes under various conditions or over time, providing insights into how plants regulate immune responses at the transcriptional level during stress or pathogen attacks [2]. By examining the expression patterns of thousands of genes across time and varying conditions, transcriptomics allows researchers to uncover the complex regulatory networks involved in plant immune responses. This molecular-level understanding is critical for horticulture, where plants frequently encounter biotic threats, as it helps improve disease resistance and crop protection. When a plant encounters a pathogen, it activates a series of defense mechanisms orchestrated by a network of genes. These mechanisms include both general immune responses and specific reactions, tailored to particular pathogens. Transcriptomics identifies genes that are upregulated or downregulated during pathogen attacks, offering valuable insights into the pathways and molecular processes involved in the plant's immune response. By analyzing gene expression changes at various stages of infection, transcriptomics reveals how plants perceive pathogens, activate defense genes, and regulate their immune signaling networks.

In horticulture, the use of transcriptomics is essential for identifying critical genes associated with disease resistance. This knowledge aids in developing crops with enhanced pathogen resistance, thereby improving agricultural productivity and sustainability. Additionally, transcriptomics data can inform breeding programs by pinpointing genes vital for plant defense, enabling the creation of disease-resistant plant varieties. By leveraging transcriptomics data, researchers can contribute to improved plant health, enhanced food security, and the development of sustainable agricultural practices.

The physics underlying omics data is largely unknown, which means that mathematical differential equations precisely describing the system are not predefined. This necessitates the development of foundational analytical elements from scratch. The primary focus of the project is to design neural network models capable of handling the time-dependent nature of the data. To begin addressing the challenge of undefined physical systems, simplified ordinary differential equations (ODEs) have been proposed for a specific part of the data. These equations serve as an approximation of the temporal interactions within specific parts of the dataset, enabling the PINNs to focus on capturing sequential relationships and estimating parameters. By training PINNs on subsets of the time-series transcriptomics data, the project explores how these networks can be adapted to high-dimensional biological datasets and used to uncover temporal dependencies and predict the results. [12] [8].

In physics-informed machine learning, Neural ODEs play a crucial role in incorporating dynamic constraints directly into learned models. When combined with approaches like Physics-Informed Neural Networks (PINNs) or Physics-Informed Variational Autoencoders (ϕ-DVAE), they allow researchers to enforce physical laws while benefiting from the flexibility of deep learning. This makes Neural ODEs especially valuable for studying biological systems, where understanding temporal dependencies is essential for example, in transcriptomics or modeling molecular interactions in plant-pathogen dynamics.

4 Related Works

4.1 Physics-Informed Neural Networks (PINNs) for Data-Driven Inverse Problems

Physics-Informed Neural Networks (PINNs) represent a powerful tool for solving data-driven inverse problems, especially in situations where the governing Partial Differential Equations (PDEs) are either unknown or only partially understood. By integrating neural networks with fundamental mathematical principles, PINNs are able to infer both the hidden dynamics and the unknown parameters directly from observed data. This is achieved by leveraging automatic differentiation, which allows the network to approximate solutions and their derivatives, thus enabling the discovery of the underlying equations while adhering to physical constraint [9]s.

The fundamental concept behind PINNs is to assume a general form for the PDE and then combine data-driven loss functions with residual terms that ensure consistency with the physical laws governing the system. The residual terms quantify the discrepancy between the model's predictions and the physical constraints imposed by the PDE. During the training process, the PINNs minimize a total loss function, which includes both observed data and the residuals of the unknown governing equations. This approach allows the model to simultaneously learn the solution to the problem and uncover the structure of the PDE, including any unknown terms and parameters [11] [4]. This flexibility makes PINNs especially effective when dealing with noisy or incomplete data and provides a unified framework for addressing challenges such as parameter estimation, missing boundary conditions, and discovering nonlinear operators in PDEs.

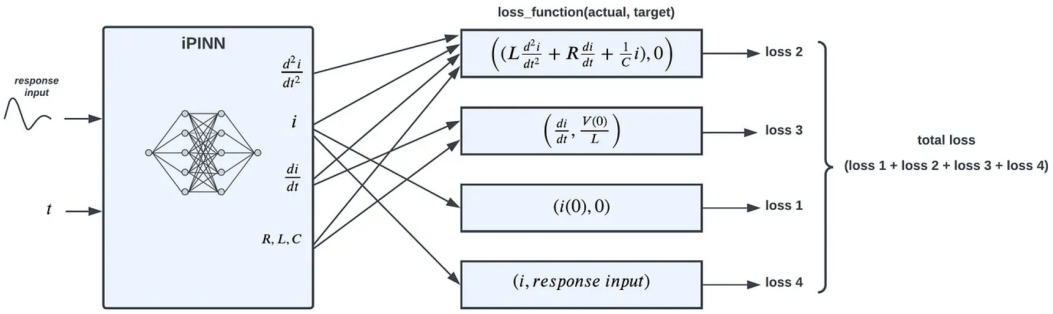

Fig. 1: Physics-Informed Neural Networks (PINNs).[14]

To begin with, the general form of the PDE is assumed without prior knowledge of the specific terms. This equation is typically expressed as:

$$\mathcal{N}[u; \lambda] = 0,$$

where $u(x, t)$ is the solution to the PDE, and $\mathcal{N}[u; \lambda]$ represents a nonlinear operator parameterized by unknown coefficients λ. These coefficients can correspond to various physical properties, such as reaction rates, diffusion coefficients, or other system parameters. The form of $\mathcal{N}$ might include terms involving derivatives, nonlinear reactions, or diffusion processes, but the exact nature of these terms is typically unknown at the start.

In the PINN framework, a neural network, $N(x, t)$, is used to approximate the solution $u(x, t)$. The network is designed to approximate not just the solution itself but also its derivatives. Since automatic differentiation allows for the efficient calculation of derivatives such as u_t, u_x, and u_{xx}, the neural network can compute the necessary quantities to construct the governing equations.

To enforce the residuals of the assumed PDE, a physics-based loss function is defined. The physics loss function can be expressed as:

$$f(x, t) = N_t + \mathcal{N}[N; \lambda].$$

This function ensures that the output of the neural network adheres to the governing PDE, thereby incorporating the underlying physics into the learning process.

During the training of the PINN, two primary loss functions are minimized: the data loss and the physics loss. The data loss quantifies the difference between the predicted values from the neural network and the observed data, and it is defined as:

$$L_{\text{data}} = \frac{1}{N} \sum_{i=1}^{N} \left| N(x_i, t_i) - u_{\text{true}}(x_i, t_i) \right|^2,$$

where $u_{\text{true}}(x, t)$ is the true observed data at the points (x_i, t_i). The data loss function ensures that the model's predictions are as close as possible to the observed data, providing a means of training the model with real-world information.

The physics loss, on the other hand, minimizes the residual of the physics-based loss, and is given by:

$$L_{\text{physics}} = \frac{1}{M} \sum_{j=1}^{M} \left| f(x_j, t_j) \right|^2,$$

where M represents the number of collocation points sampled within the domain of the PDE. This term ensures that the solution produced by the neural network is consistent with the underlying physics of the problem, enforcing the structure of the PDE during the learning process.

The total loss function for the PINN is then the sum of the data loss and the physics loss:

$$L = L_{\text{data}} + L_{\text{physics}}.$$

This combined loss function enables the neural network to learn both the solution to the problem and the structure of the governing PDE, thereby addressing the inverse problem in a unified manner.

As the PINN is trained, it not only learns the solution to the PDE but also infers the unknown parameters λ, which are the coefficients of the governing equations. These parameters may represent physical quantities such as

material properties, reaction rates, or other system parameters that are critical for determining the system's behavior. In addition to learning the coefficients, the PINN framework also uncovers the form of the nonlinear operator $\mathcal{N}[u;\lambda]$, providing valuable insights into the physical processes governing the system.

PINNs are particularly well-suited for biological systems, geophysics, fluid dynamics, and material science, where the physics of the system is often complex or partially understood [13]. By bridging the gap between data and scientific modeling, PINNs enable the discovery of interpretable and physically consistent equations, offering a powerful tool for advancing scientific research in domains reliant on sparse, noisy, or incomplete datasets.

4.2 (ϕ-DVAE): Physics-Informed Dynamical Variational Autoencoders

The ϕ-DVAE framework integrates variational autoencoders (VAEs) with physics-informed modeling to bridge the gap between unstructured data and systems governed by partial differential equations (PDEs) or stochastic differential equations (SDEs). This innovative approach combines machine learning with physical laws, enabling the assimilation of noisy and sparse data into complex dynamical systems. Unlike traditional models, ϕ-DVAE preserves physical consistency while inferring latent dynamics and estimating unknown parameters, making it a powerful tool for state and parameter estimation.

A key feature of ϕ-DVAE is the embedding of latent state dynamics within a physics-constrained framework. These dynamics are governed by ordinary differential equations (ODEs), PDEs, or SDEs, providing physical realism to the latent space. In contrast to standard VAEs, which operate in an unconstrained latent space, this incorporation of physical laws adds a layer of reliability and interpretability to the generative process.

The latent dynamics in ϕ-DVAE follow stochastic differential equations (SDEs), which capture the continuous evolution of the system, accounting for both deterministic and stochastic influences. To achieve this, the framework employs statistical finite element methods (statFEM) to discretize the SDEs, enabling effective integration of physical models with machine learning.

The generative process in ϕ-DVAE transforms high-dimensional unstructured data, such as videos or images, into a low-dimensional latent space using the VAE encoder. These latent states evolve over time according to the latent dynamics described by SDEs. The decoder then reconstructs observations probabilistically from the latent states, modeling the relationship between noisy high-dimensional data and their corresponding latent representations.

One of the significant strengths of ϕ-DVAE is its ability to perform joint inference of latent states, unknown parameters governing the physical dynamics, and neural network parameters (encoder and decoder). This is achieved through variational inference and Monte Carlo sampling, providing a probabilistic framework for parameter estimation and uncertainty quantification.

Uncertainty quantification is another critical aspect of ϕ-DVAE. By using variational Bayesian methods, the framework estimates the posterior distribution of both latent states and parameters, capturing the uncertainty associated with the underlying physical model. Additionally, ϕ-DVAE employs Extended Kalman Filtering (ExKF) to infer latent states from pseudo-observations, ensuring accurate state inference even in the presence of noisy or incomplete data.

The mathematical foundation of ϕ-DVAE relies on probabilistic relationships between latent dynamics, observations, and pseudo-observations. The latent dynamics are modeled as:

$$u_n|u_{n-1},\Lambda \sim p(u_n|u_{n-1},\Lambda),$$

where u_n represents the latent states, Λ denotes unknown parameters in the physical model, and $p(u_n|u_{n-1},\Lambda)$ describes the transition dynamics.

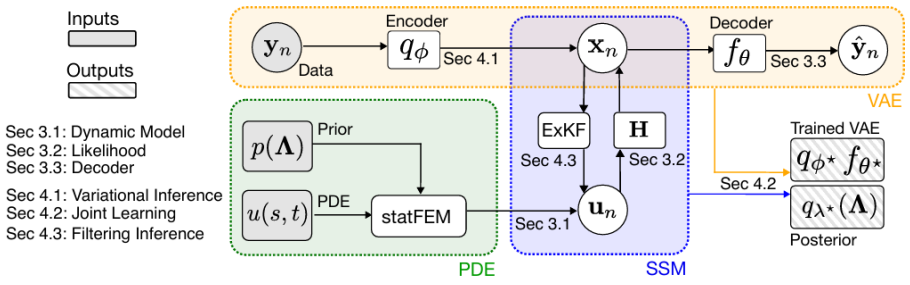

Fig. 2: Flow diagram describing connections between the specified partial differential equation (PDE), latent state-space model (SSM) and variational autoencoder (VAE).[7]

Observations, such as noisy video frames (y_n), are generated from the latent states through a probabilistic decoder:

$$y_n|x_n \sim p_\theta(y_n|x_n),$$

where x_n are the pseudo-observations, and $p_\theta(y_n|x_n)$ represents the likelihood function.
The mapping between latent states and pseudo-observations is given by:

$$x_n = Hu_n + r_n, \quad r_n \sim \mathcal{N}(0, R),$$

where H is the observation matrix, and r_n is Gaussian noise.
To approximate the true posterior distribution of the latent states and parameters, ϕ-DVAE employs a variational posterior:

$$q(u_{1:N}, x_{1:N}, \Lambda|y_{1:N}) = q(u_{1:N}|x_{1:N}, \Lambda)q_\phi(x_{1:N}|y_{1:N})q_\lambda(\Lambda),$$

where $q_\phi(x_{1:N}|y_{1:N})$ maps data to pseudo-observations, and $q_\lambda(\Lambda)$ provides the variational approximation of the model parameters.
The model is trained by maximizing the Evidence Lower Bound (ELBO):

$$\log p(y_{1:N}) \geq E_{q_\phi}\left[\log p_\theta(y_{1:N}|x_{1:N}) - \log q_\phi(x_{1:N}|y_{1:N})\right] + E_{q_\lambda}\left[\log p(x_{1:N}|\Lambda) + \log p(\Lambda) - \log q_\lambda(\Lambda)\right].$$

The ϕ-DVAE framework has shown great versatility in modeling both linear and nonlinear dynamical systems. It has been successfully applied to a variety of systems, including the advection equation, the Lorenz-63 system, and the Korteweg–de Vries (KdV) equation. In these applications, ϕ-DVAE has outperformed traditional methods like Kalman VAE (KVAE), accurately estimating unknown parameters and predicting system behavior even in the presence of noisy data. This demonstrates its ability to effectively integrate physical laws with machine learning for robust system identification and prediction.

4.3 Neural Ordinary Differential Equations

Neural Ordinary Differential Equations (Neural ODEs), introduced by Chen et al. (2018), represent a novel class of deep learning models that generalize traditional neural networks by parameterizing the dynamics of hidden states using continuous-time differential equations.

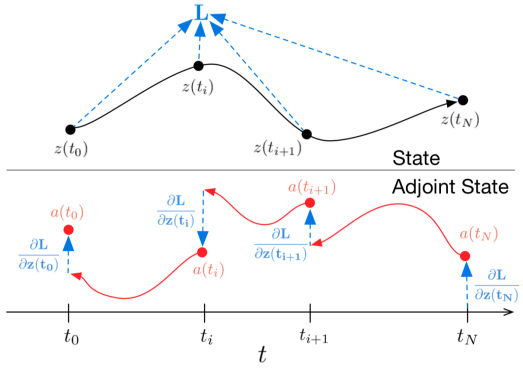

Fig. 3: Reverse-mode differentiation of an ODE solution. The adjoint sensitivity method solves an augmented ODE backward in time.[3]

$$\frac{d\mathbf{h}(t)}{dt} = f(\mathbf{h}(t), t, \theta),$$

where f is a neural network with parameters θ and t denotes continuous time. The state evolution is computed via an ODE solver that adaptively selects the number of function evaluations based on the task's complexity. This continuous formulation naturally accommodates irregularly sampled data—a frequent challenge for traditional recurrent neural networks (RNNs) and long short-term memory networks (LSTMs)[15] [1].

In time series analysis, the input sequence $\mathbf{x}(t)$ is first mapped to a latent representation $\mathbf{h}(t)$ using an encoder. The dynamics of this latent state are then modeled as

$$\mathbf{h}(t_1) = \mathbf{h}(t_0) + \int_{t_0}^{t_1} f(\mathbf{h}(t), t, \theta)\, dt,$$

where t_0 and t_1 are the start and end times of the interval, respectively. The integral is computed using an ODE solver, which provides flexibility in handling irregular time intervals.

A significant advantage of Neural ODEs is their memory efficiency, achieved through the adjoint sensitivity method, which computes gradients without storing intermediate states. In addition, Neural ODEs can be extended to model stochastic dynamics by incorporating noise terms, leading to Neural Stochastic Differential Equations (SDEs) that capture inherent uncertainties in the data.

An important development in this field is the integration of Neural ODEs with latent variable models for time series analysis. In this framework, an RNN encoder infers an initial latent state z_{t_0} from irregular observations $\{x_{t_i}\}$. This state is then propagated through time using an ODE solver, resulting in a latent trajectory $z_{t_1}, \ldots, z_{t_n}$. The generative process is described by the following equations

A local initial state z_{t_0} sampled from a prior distribution:

$$z_{t_0} \sim p(z_{t_0})$$

A global set of latent dynamics shared across all time series, defined by an ODE:

$$z_{t_N} = \text{ODESolve}(z_{t_0}, f_\theta, t_0, t_N)$$

Observations generated from the latent state at each time point:

$$x_{t_i} \sim p(x|z_{t_i}, \phi_x)$$

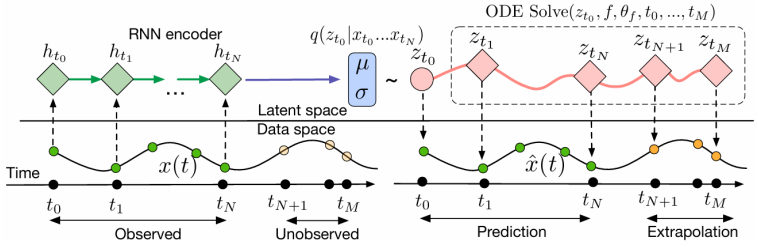

Fig. 4: Computation graph of the latent ODE model.[3]

This approach, exemplified by the work of Rubanova et al. (2019) on Latent ODEs, has shown superior performance over traditional RNN methods, particularly for tasks requiring seamless interpolation and extrapolation of time series data.

The versatility of Neural ODEs and their extensions make them a valuable tool in a wide range of applications. Their ability to model continuous dynamics and handle irregular sampling has been effectively applied in healthcare for monitoring patient data with irregular lab measurements, in finance for predicting stock prices in volatile markets, and in physics-informed modeling where they are combined with stochastic differential equations to enforce biophysical constraints.

5 Methodology

The project focuses on the analysis of time-course transcriptomics data from tomato plants infected by three distinct pathogens, with data collected at various time points for both leaves and roots. This dataset is crucial for studying the dynamic mechanisms of disease resistance and plant responses. While the dataset does not have missing values, it presents significant analytical challenges due to its high sparsity, consisting of 3 times points and approximately 7 genes, along with a limited number of samples. These characteristics make traditional analytical approaches inadequate for capturing the temporal and biological complexities inherent in the data

To address these challenges, the project employs advanced computational methods, specifically Physics-Informed Neural Networks (PINNs) and Neural ODE. These methods integrate observational data with physical or mathematical models to provide a robust framework for analyzing biological dynamics, mitigating batch effects, and uncovering interactions within plant defense mechanisms over time.

Given our observed experimental data, we employed ordinary differential equations (ODEs) rather than partial differential equations (PDEs) for model fitting, as the temporal dynamics of our system could be sufficiently captured by ODE formulations. We specifically adopted the first-order linear ODE model:

$$\frac{dy}{dt} = \alpha + \beta \cdot y(t) + \gamma \cdot t \tag{1}$$

where α, β, and γ represent the system parameters to be estimated.

Our analysis framework incorporated three distinct computational approaches.

The first approach, Inverse Physics-Informed Neural Networks (Inverse PINN), combines neural networks with physics-based constraints to estimate the ODE parameters directly. A neural network was trained to approximate the solution of the ODE while enforcing the physics of the system through a custom loss function. This loss function consisted of two components: a data loss term, which measured the mean squared error (MSE) between the predicted and observed gene expression values, and a physics loss term, which penalized deviations from the ODE by ensuring that the derivative of the predicted solution (computed via automatic differentiation) satisfied the ODE. By minimizing this combined loss, we estimated the parameters α, β, and γ for each gene.

The neural network architecture consisted of three layers (two hidden layers and one output layer) with 50 neurons in each hidden layer and Tanh activation functions. This compact architecture was chosen to maintain computational efficiency and avoid overfitting given the limited dataset.

The second approach utilized the Neural ODE framework, which provides a data-driven method for solving differential equations by parameterizing the ODE function with a neural network. For each gene, a neural network was trained to directly model the ODE dynamics, and the parameters α, β, and γ were learned by minimizing the MSE between the predicted and observed gene expression values. This approach leverages the flexibility of neural networks to capture complex dynamics. The architecture for this framework consisted of two hidden layers with 16 neurons each, using Tanh activation functions. The model was trained using the adjoint sensitivity method, which enables efficient gradient computation for continuous-depth models.

Finally, We developed a **Hybrid Approach (Neural ODE + Inverse PINN) that combined the automatic differentiation capabilities of Neural ODEs with the physics constraints of inverse PINNs** This method combined prior knowledge of the ODE structure with the flexibility of neural networks. The combined loss function included a data loss term (MSE between the predicted and observed gene expression values) and a physics loss term (penalizing deviations from the ODE). This hybrid approach enabled simultaneous learning of both system dynamics and physical parameters, providing a robust framework for parameter estimation. The architecture used for this method was similar to the Neural ODE framework, with two hidden layers and 16 neurons per layer, ensuring simplicity and avoiding overfitting.

Due to the limited size of our dataset, we maintained simple neural network architectures across all approaches to avoid overfitting while ensuring adequate model expressivity.

6 Result and Discussion

The ODE model Equation 1 was chosen for its simplicity and ability to capture key dynamic behaviors in gene expression. Using this model, we trained and evaluated three different approaches: Inverse Physics-Informed Neural Networks (Inverse PINN), Neural ODE, and a Hybrid Model that combines the strengths of both methods.

The Inverse PINN model was trained on six distinct gene expressions of transcriptomics datasets. For each dataset, the model estimated the parameters (α, β, γ) by minimizing a combined loss function that incorporates both data loss (mean squared error between observed and predicted values) and physics loss (penalty for deviations from the ODE). The learned parameters were then used to cluster the datasets into four distinct groups based on their dynamic behaviors. These clusters revealed genes with similar regulatory mechanisms and responses to experimental conditions.

We repeated the analysis for all three models—Inverse PINN, Neural ODE, and the Hybrid Model (Inverse PINN + Neural ODE)—and evaluated their performance in terms of parameter estimation, and loss minimization. The results are summarized in the below figures 5, 6, 7, 8, 9, 10, 11, 11, 12, 13.

Learned parameters with cluster assignments:

	alpha	beta	gamma	gene_id	cluster
0	0.196849	1.436124	0.443261	AT5G40100	2
1	0.207240	1.725898	0.097130	AT1G06930	1
2	0.177277	1.518552	0.210680	AT1G04470	0
3	0.291859	1.595881	0.292140	AT5G66730	1
4	0.067159	1.665267	0.018560	AT2G38472	3
5	0.228233	1.613020	0.226064	AT2G38474	1
6	0.200077	1.521567	0.159467	AT2G38470	0

Fig. 5: parameters and the clusters of the genes the case of IPINN.

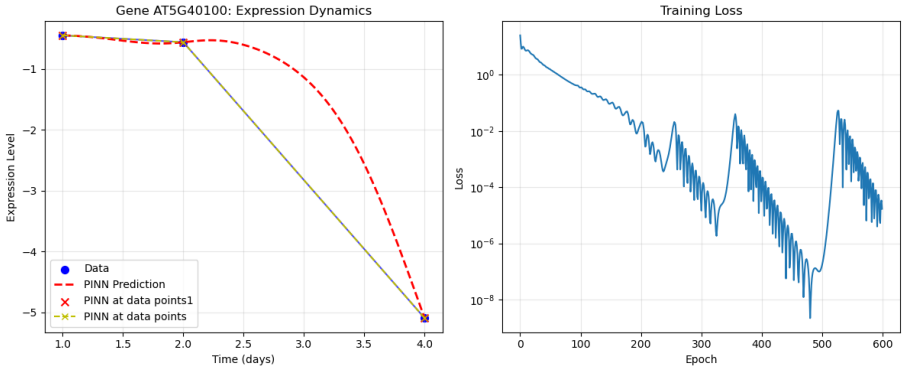

Fig. 6: For Gene AT5G40100, the left graph shows the observed data, predicted data, and model predictions, while the right graph displays the training loss curve in the case of IPINN.

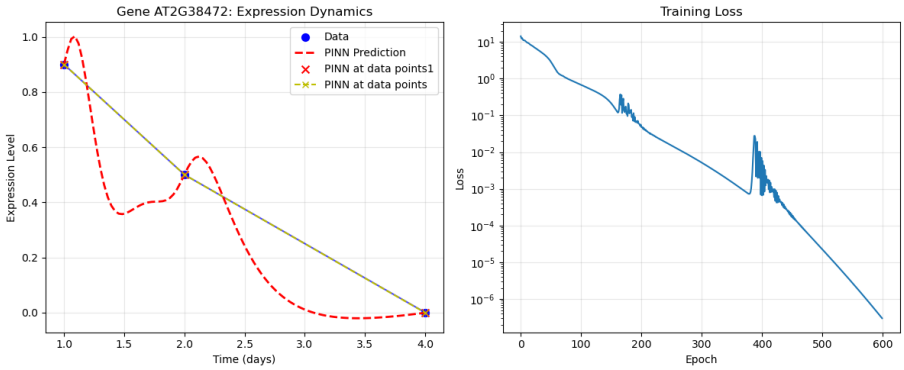

Fig. 7: For Gene AT2G38472, the left graph shows the observed data, predicted data, and model predictions, while the right graph displays the training loss curve in the case of IPINN.

Parameters with Cluster Assignments:

	Gene	alpha	beta	gamma	cluster
0	AT5G40100	-0.114982	0.456523	-0.180553	2
1	AT1G06930	-0.287002	-0.350527	-0.075282	1
2	AT1G04470	0.820759	-0.229800	-0.269294	0
3	AT5G66730	0.027203	-0.051694	-0.029655	1
4	AT2G38472	-0.107584	-0.125984	-0.058369	1
5	AT2G38474	0.325214	-0.095020	-0.113117	3
6	AT2G38470	0.006043	-0.017935	0.004196	1

Fig. 8: parameters and the clusters of the genes in the case of Neural ODE.

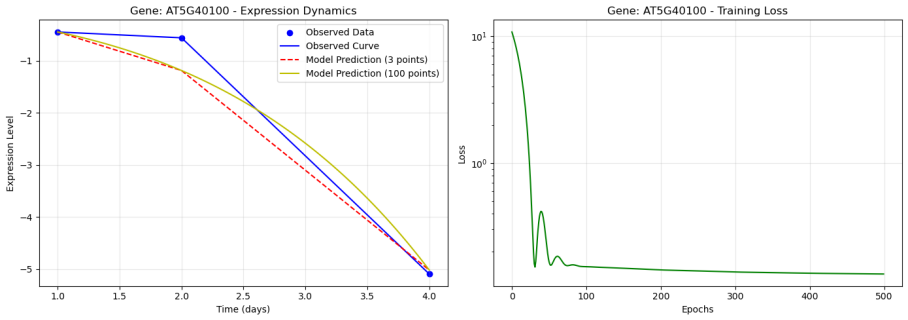

Fig. 9: For Gene AT5G40100, the left graph shows the observed data, predicted data, and model predictions, while the right graph displays the training loss curve in the case of Neural ODE.

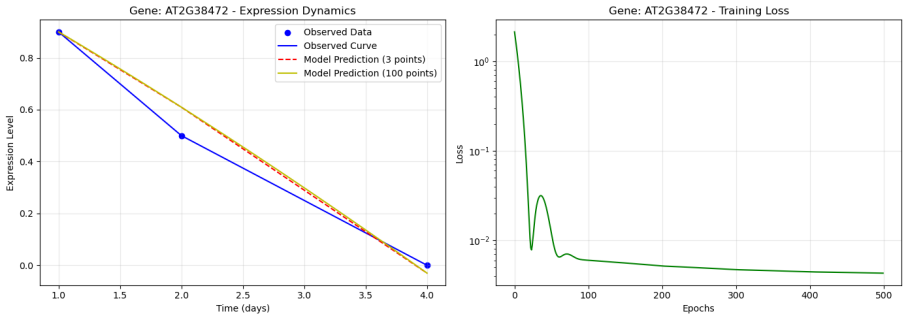

Fig. 10: For Gene AT2G38472, the left graph shows the observed data, predicted data, and model predictions, while the right graph displays the training loss curve in the case of Neural ODE.

Cluster assignments for each gene:

	Gene	alpha	beta	gamma	cluster
0	AT5G40100	-0.003555	0.364435	-0.286588	2
1	AT1G06930	-0.205956	-0.302995	-0.097479	1
2	AT1G04470	0.690664	-0.400836	-0.205206	0
3	AT5G66730	0.036836	-0.049357	-0.033600	1
4	AT2G38472	-0.047874	-0.088919	-0.084477	1
5	AT2G38474	0.258217	-0.141527	-0.087192	3
6	AT2G38470	0.008556	-0.015947	0.002505	1

Fig. 11: parameters and the clusters of the genes in the case of the hybrid model(IPINN+Neural ODE).

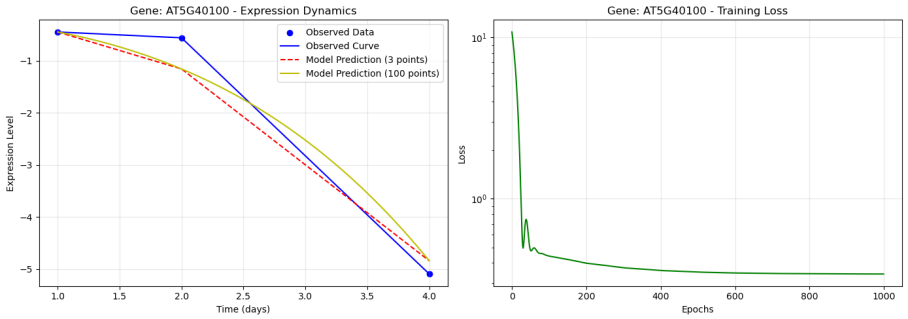

Fig. 12: For Gene AT5G40100, the left graph shows the observed data, predicted data, and model predictions, while the right graph displays the training loss curve in the case of the hybrid model(IPINN+Neural ODE).

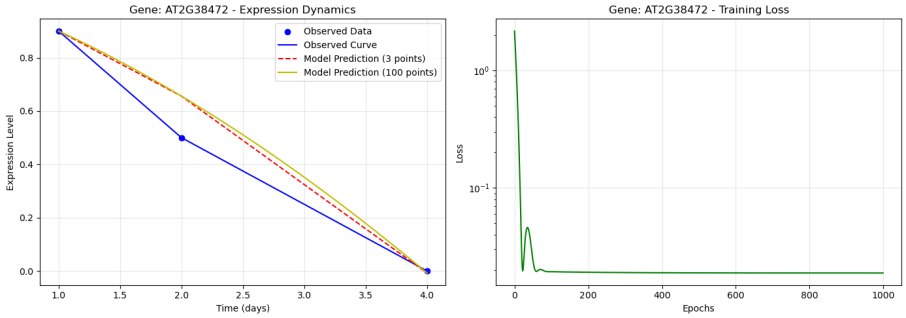

Fig. 13: For Gene AT2G38472, the left graph shows the observed data, predicted data, and model predictions, while the right graph displays the training loss curve in the case of the hybrid model(IPINN+Neural ODE).

6.1 Comparison of Models

To validate the model, we used test data that followed the dynamics of Cluster 2 from the training dataset. Using the parameter values (α, β, γ) learned for Cluster 2, we predicted the behavior of the test data and compared it to the observed values. The predictions showed good agreement for the first two data points but deviated for the last data point, indicating potential limitations in the model's ability to generalize to certain regions of the parameter space.

We repeated the analysis for all three models—Inverse PINN, Neural ODE, and the Hybrid Model (Inverse PINN + Neural ODE)—and evaluated their performance in terms of parameter estimation. The results are summarized below in the following figures : 14, 15, 16.

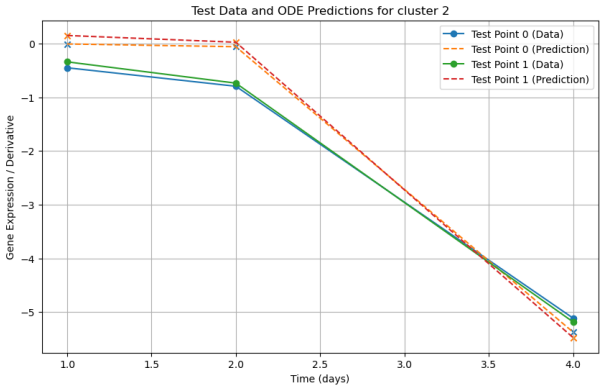

Fig. 14: For test data, predicted data in the case of the IPINN.Cluster 2 ($\alpha = 0.196849$, $\beta = 1.436124$, $\gamma = 0.443261$)

6.2 Insights from Clustering and Parameter Analysis

The clustering of gene expression dynamics based on the learned parameters (α, β, γ) revealed distinct groups of genes with shared regulatory mechanisms. These clusters provide a mechanistic understanding of gene behavior under experimental conditions, highlighting the utility of the ODE model in transcriptomics data analysis. For example, α represents the basal transcription or degradation rate, independent of the current expression level. β captures feedback effects, with positive values indicating self-activation and negative values indicating self-inhibition. γ accounts for time-dependent external influences, such as environmental or experimental factors.

7 Conclusion

This study demonstrates the utility of combining physics-informed machine learning approaches with mechanistic ODE models for transcriptomics data analysis. The Inverse PINN, Neural ODE, and Hybrid Model each offer unique advantages, with the Hybrid Model providing a balance between interpretability and flexibility. By bridging the gap

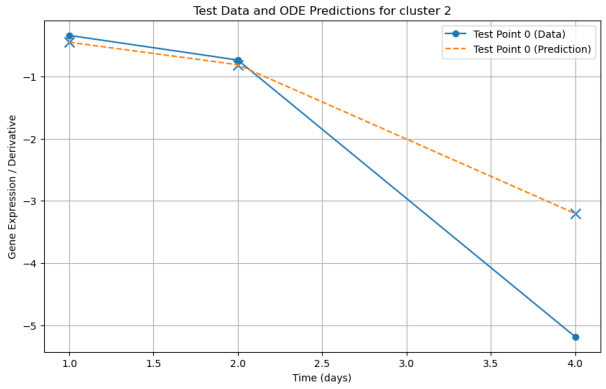

Fig. 15: For test data, predicted data in the case of the Neural ODE.Cluster 2 ($\alpha = -0.114982$, $\beta = 0.456523$, $\gamma = -0.180553$)

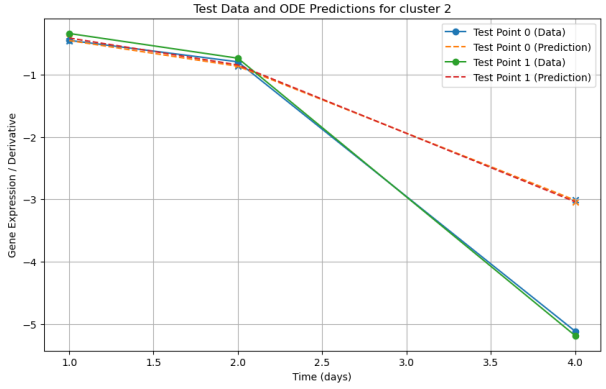

Fig. 16: For test data, predicted data in the case of the (IPINN+Neural ODE).Cluster 2 ($\alpha = -0.003555$, $\beta = 0.364435$, $\gamma = -0.286588$)

between purely statistical methods and mechanistic models, this approach offers both predictive power and biological interpretability, paving the way for more advanced applications in systems biology and gene regulation studies.

8 Limitations and Future Work

While the results demonstrate the potential of physics-informed machine learning approaches in modeling gene expression dynamics, several limitations were identified. The limited number of time points in the dataset constrains the model's ability to capture finer details of the dynamics. Future studies should include more time points to better constrain the model. The deviations observed in certain test cases suggest that the models may struggle to generalize to regions of the parameter space not well-represented in the training data. Incorporating gene-gene interactions through coupled ODEs could provide a more comprehensive understanding of regulatory networks. Additionally, combining transcriptomics data with other omics data types (e.g., proteomics, metabolomics) could enable multi-scale modeling of gene regulation.

.

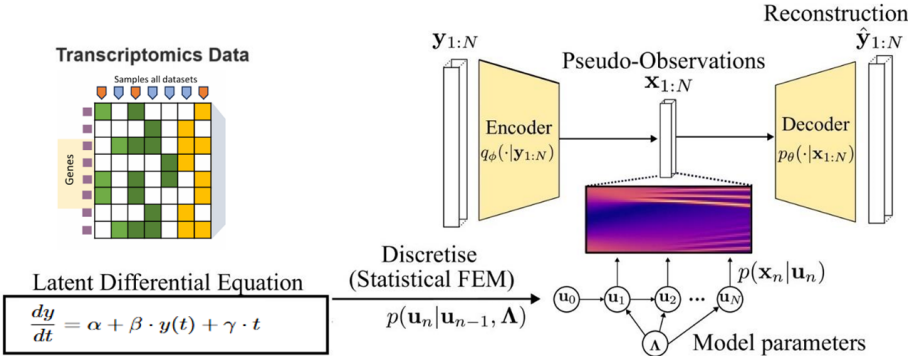

Fig. 17: The architecture of the Physics-Informed Dynamical Variational Autoencoder is being developed for time-series transcriptomics data analysis. We have some, governing ordinary differential equations (ODEs), and efforts are focused on defining the neural network for the model.

In addition to the use of PINNs and neural ODEs, in the future, we will incorporates Dynamical Variational Autoencoders (DVAEs) to process sequential data. DVAEs extend traditional Variational Autoencoders (VAEs) by combining them with temporal models, often incorporating state-space models or recurrent neural networks[6] [10]. This allows for unsupervised representation learning tailored to sequential data, enabling the generation of latent vectors that encode the essential features and temporal interactions of the transcriptomics dataset. These latent representations provide a reduced-dimensional yet comprehensive view of the data, capturing critical elements relevant to plant defense mechanisms and their responses to pathogen infections. Here we already introduced a simplified ODE in equation 1. Also, we can integrate simplified PDEs from the part of omics data to introduce physics into the DVAE (ϕ-DVAE). These state-of-the-art techniques combine observational data with physical or mathematical models, enabling the capture of complex biological dynamics and interactions. The proposed methodology focuses on removing batch effects and uncovering temporal dependencies in transcriptomic data to reveal the underlying patterns of plant-pathogen interactions.

References

1. Sören Becker, Michal Klein, Alexander Neitz, Giambattista Parascandolo, and Niki Kilbertus. Discovering ordinary differential equations that govern time-series. *arXiv preprint arXiv:2211.02830*, 2022.
2. Yunpeng Cao, Xiaoxu Li, Hui Song, Muhammad Abdullah, and Muhammad Aamir Manzoor. Multi-omics and computational biology in horticultural plants: from genotype to phenotype, volume ii, 2024.
3. Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. *Advances in neural information processing systems*, 31, 2018.
4. Salvatore Cuomo, Vincenzo Schiano Di Cola, Fabio Giampaolo, Gianluigi Rozza, Maziar Raissi, and Francesco Piccialli. Scientific machine learning through physics–informed neural networks: Where we are and what's next. *Journal of Scientific Computing*, 92(3):88, 2022.

5. Rossin Erbe, Genevieve Stein-O'Brien, and Elana J Fertig. Transcriptomic forecasting with neural ordinary differential equations. *Patterns*, 4(8), 2023.
6. Laurent Girin, Simon Leglaive, Xiaoyu Bie, Julien Diard, Thomas Hueber, and Xavier Alameda-Pineda. Dynamical variational autoencoders: A comprehensive review. *arXiv preprint arXiv:2008.12595*, 2020.
7. Alex Glyn-Davies, Connor Duffin, O Deniz Akyildiz, and Mark Girolami. ϕ-dvae: Physics-informed dynamical variational autoencoders for unstructured data assimilation. *Journal of Computational Physics*, 515:113293, 2024.
8. Paguiel Javan Hossie, Béatrice Laroche, Thibault Malou, Lucas Perrin, Thomas Saigre, and Lorenzo Sala. Simulating interactions in microbial communities through physics informed neural networks: towards interaction estimation. 2024.
9. George Em Karniadakis, Ioannis G Kevrekidis, Lu Lu, Paris Perdikaris, Sifan Wang, and Liu Yang. Physics-informed machine learning. *Nature Reviews Physics*, 3(6):422–440, 2021.
10. Diederik P Kingma, Max Welling, et al. An introduction to variational autoencoders. *Foundations and Trends® in Machine Learning*, 12(4):307–392, 2019.
11. Isaac E Lagaris, Aristidis Likas, and Dimitrios I Fotiadis. Artificial neural networks for solving ordinary and partial differential equations. *IEEE transactions on neural networks*, 9(5):987–1000, 1998.
12. Lu Lu, Xuhui Meng, Zhiping Mao, and George Em Karniadakis. Deepxde: A deep learning library for solving differential equations. *SIAM review*, 63(1):208–228, 2021.
13. Chuizheng Meng, Sungyong Seo, Defu Cao, Sam Griesemer, and Yan Liu. When physics meets machine learning: A survey of physics-informed machine learning. *arXiv preprint arXiv:2203.16797*, 2022.
14. 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.
15. Yulia Rubanova, Ricky TQ Chen, and David K Duvenaud. Latent ordinary differential equations for irregularly-sampled time series. *Advances in neural information processing systems*, 32, 2019.
16. Jeyachandran Sivakamavalli and Baskaralingam Vaseeharan. An overview of omics approaches: Concept, methods and perspectives. 2020.
17. Xiaoyu Zhang, Jingqing Zhang, Kai Sun, Xian Yang, Chengliang Dai, and Yike Guo. Integrated multi-omics analysis using variational autoencoders: application to pan-cancer classification. In *2019 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pages 765–769. IEEE, 2019.