Ridwan Ademola, IBRAHIM

Physics – Constrained Neural Network for the Spatio-temporal Prediction and Reconstruction of the Tilt and Deflection of Smart Bridges.

School of Technology and Innovations
Smart Grid
Smart Energy

Vaasa 2026

UNIVERSITY OF VAASA
School of Technology and Innovations
Author: Ridwan Ademola, IBRAHIM
Title of the thesis: Physics – Constrained Neural Network for the Spatio-temporal Prediction and Reconstruction of the Tilt and Deflection of Smart Bridges.
Degree: Master of Sciences
Degree Programme: Smart Energy
Supervisor: Prof. Stavros G. Stavrinides
Year: 2026 **Pages:** 89

ABSTRACT:

Smart cities increasingly rely on large-scale Internet of Things (IoT) networks to monitor critical infrastructure (such as bridges) and support data-driven decision making. Monitoring the health of bridges requires accurate information on the structural responses - such as tilt and deflection - at critical locations. This information is usually gathered through distributed sensing systems or discrete point sensors. Distributed sensors provide spatially high-resolution information but also generate large volumes of data, increase communication and maintenance costs, and create scalability challenges for long-term monitoring systems. In contrast, discrete sensors can only provide information at their installation positions.

This thesis presents a framework that uses only one discrete sensor per span to reconstruct the longitudinal tilt and deflection of a continuous prestressed concrete girder at uninstrumented positions. The proposed framework integrates autoregressive forecasting and Physics-Constrained Neural Networks (PCNNs) based on Euler–Bernoulli beam theory to address inverse problems associated with uncertain loads and material properties arising from construction imperfections.

Four interconnected models are developed: (i) an Autoregressive (AR) model for multi-day sensor signal forecasting, (ii) a PCNN model that combines sensor measurements with environmental features to reconstruct full-span tilt and deflection responses, (iii) an integrated AR-PCNN pipeline for predictive full-span reconstruction, and (iv) a Finite Element Method (FEM) model used as a baseline for validating the physics-constrained framework. The models are evaluated using real-world data from the IDA-KI OpenLab research bridge. Results show that the AR-PCNN framework achieved an R^2 of up to 0.9 while maintaining PDE residual below $4.9\text{x}10^{-4}$, demonstrating strong accuracy and physical consistency.

KEYWORDS: Structural Health Monitoring; Physics-Constrained Neural Networks; Euler–Bernoulli Beam Theory; Inverse Problems; Finite Element Method; Autoregression, Sparse Sensor Networks; Prestressed Concrete Girder

Acknowledgement

I would like to express my deepest gratitude to Allah (SWT) for His mercy, strength, guidance, and blessings throughout this journey. I acknowledge myself for the consistency, self-discipline, and belief that carried me through this demanding academic journey. My sincere appreciation goes to my parents and siblings for their unwavering support, prayers, sacrifices, and encouragement throughout this program. I am equally grateful to my fiancée, Omolade, for her patience, understanding, and emotional support during the two years I spent away from Nigeria pursuing this degree.

I am profoundly grateful to the Erasmus Mundus SMACCs program for this life-changing opportunity and for the invaluable academic and cultural experiences gained throughout the program. I would like to sincerely thank my supervisor, Prof. Stavros Stavrinides, for his continuous guidance, encouragement, and support throughout this thesis. Beyond being a supervisor, he has been a mentor and a fatherly figure during this journey.

I also appreciate the coordinators across the SMACCs partner universities, including Prof. Luis Portilla, Prof. Christos Tjortjis, Prof. Hannu Laaksonen, and Prof. Zacharie De Grève, for their support and commitment to the success of the program. Finally, I express my gratitude to Max Herbers, Bertram Richter, and all the researchers of the IDA-KI OpenLab for granting access to their bridge monitoring data and supporting this research.

Declaration

I hereby declare that this thesis is my own original work and was completed independently. All sources of information, data, and ideas obtained from the work of others have been appropriately acknowledged and referenced.

In the course of the thesis preparation, I used artificial intelligence (AI) tools in a limited and supportive capacity. ChatGPT was primarily used as a search engine to help identify and locate relevant academic literature for review. Grammarly, and in few times ChatGPT, was used to improve the clarity, grammar, and academic language of the written text. Google Gemini was equally used in generating some figures for visual illustrations. These tools were not used to generate the research methodology, analysis, results, interpretations, or conclusions presented in this thesis. Therefore, full responsibility for the content, accuracy, and integrity of the work remains entirely mine.

Contents

1 Introduction 11
1.1 Background 11
1.2 Motivation 12
1.3 Aim and Objectives 13
1.4 Research Scope 14
1.5 Limitation 14
1.6 Contribution to Knowledge 14
1.7 Structure of Thesis 15
2 Literature Review 16
2.1 Bridge Response 16
2.1.1 Tilt 16
2.1.2 Deflection 18
2.2 Forms of Analysis 19
2.2.1 Data-Driven Analysis 19
2.2.2 Finite Element Method 22
2.2.3 Physics – Informed Neural Network 23
2.2.4 Physics – Constrained Neural Network 25
2.3 Notable Reviewed Literature 30
3 Case Study 33
3.1 Site Description and Research Context 33
3.2 Structural Geometry and Material Properties 34
3.3 Support and Boundary Conditions 35
3.4 Sensor Layout 36
3.5 Loading System 37
3.6 Measured Data 38
4 Data Preprocessing 39
4.1 Data Cleaning 39
4.2 Linear Relationship 40

4.3 Feature Engineering and Transformation 41
4.4 Mutual Information 42
4.5 Transfer Entropy 43
5 Modelling 45
5.1 Autoregressive Model 46
5.1.1 Train/Test Split and Validation 47
5.2 Physics – Constrained Neural Network 47
5.2.1 Single Polynomial Trial Function Architecture 49
5.2.2 Piecewise Polynomial Trial Function Architecture 52
5.2.3 Pretrained EnvelopeNet Trial Function Architecture 52
5.3 Autoregressive-PCNN Hybrid 53
5.4 Finite Element Method 54
5.4.1 Inverse Analysis for Mean Phase Load Identification 54
5.4.2 Inverse Identification of Observation-Specific Phase Loads 56
5.4.3 Forward Analysis 56
6 Result and Discussions 57
6.1 FEM 57
6.2 PCNN 59
6.3 AR 65
6.4 AR-PCNN 68
7 Conclusions and Recommendations 70
7.1 Conclusions 70
7.1.1 Specific Conclusions 70
7.2 Recommendations 72
References 73
Appendices 78

Figures

Figure 1: Importance of Structural Health Monitoring 12
Figure 2: (a) Sisgeo 0S542HD0502 D-tiltmeter (b) Degree of freedom in space 17
Figure 3: Deflection Diagram of a Cantilever Beam 18
Figure 4: Forms of Analysis in Structural Health Monitoring 19
Figure 5: Neural Network Architecture 21
Figure 6: Physics-Informed Neural Network Architecture 24
Figure 7: IDA-KI OpenLab Bridge During Load Test Phase. 34
Figure 8: Cross-section of the girders and deck 34
Figure 9: Movable and Non-movable Joints on the Bridge 36
Figure 10: Evolution of the Static System of Span AB and BC 36
Figure 11: Sensor Layout for Tiltmeters (TIL) and Accelerometers (ACC) 37
Figure 12: Sleepers, Rail-tracks, and the Test Vehicle (Jansen et al., 2025) 37
Figure 13: Loading Phases of the Girder 38
Figure 14: Daily Averaging 39
Figure 15: Correlation Matrix 40
Figure 16: Fourier-based Features for Frequency 4 & 16 42
Figure 17: Mutual Information on the Measured Feature 43
Figure 18: Transfer Entropy 44
Figure 19: Model Chart 45
Figure 20: Single Trial Function PCNN 51
Figure 21: FEM Reconstruction Accuracy 57
Figure 22: FEM Load Discovery Per-Phase 57
Figure 23: 2D & 3D Heatmap of Full-field Slope and Deflection 58
Figure 24: Daily Slope Profile Across all Phases 58
Figure 25: Seasonal Effect on Tilt and Deflection 59
Figure 26: Model Convergence of the PCNNs 60
Figure 27: Sensor Fit at Instrumented locations for the PCNN Models 60
Figure 28: Time series of the PCNNs at Uninstrumented locations. 61
Figure 29: Validation of the PCNNs at the Uninstrumented locations 63

Figure 30: Mean Profile across full span for all PCNNs 63
Figure 31: Physics Compliance for all PCNNs 64
Figure 32: Load Discovery of the PCNNs 65
Figure 33: AR Validation Plots 66
Figure 34: AR Model Coefficients 67
Figure 35: AR Residual Analysis 68
Figure 36: AR-PCNN vs PETF Performance Analysis 68
Figure 37: AR-PCNN vs PETF Result 69
Figure 38: Piece-wise Trial Function PCNN 80
Figure 39: Pre-trained EnvelopeNet Trial Function PCNN 82
Figure 40: AR-PCNN Hybrid 84
Figure 41: Ablation Convergence Plot 87
Figure 42: Ablation Sensor Fit 88

Tables

Table 1: Summary Table of Notable Reviewed Literature 32
Table 2: Material Properties of the mid girder 35
Table 3: Frequency Breakdown 42
Table 4: AR Feature Table 46
Table 5: PCNNs Feature 49

Abbreviations

Abbreviation	Full Meaning
AR	Autoregression
AR-PCNN	AutoRegressive-Physics Constrained Neural Network
BCs	Boundary Conditions
FEM	Finite Element Method
IDA-KI	Infrastructure Data Analysis with Artificial Intelligence
L-BFGS	Limited Memory – Broyden Fletcher Goldfarb Shanno
MAE	Mean Absolute Error
MSE	Mean Squared Error
NN	Neural Network
PC	Prestressed Concrete
PCNN	Physics-Constrained Neural Network
PDEs	Partial Differential Equations
PETF	Pretrained EnvelopeNet Trial Function
PINN	Physics-Informed Neural Network
PPTF	Piecewise Polynomial Trial Function
RC	Reinforced Concrete
RMS/RMSE	Root Mean Square / Root Mean Square Error
SHM	Structural Health Monitoring
SPTF	Single Polynomial Trial Function

1 Introduction

Smart bridge monitoring systems increasingly rely on sensor technologies to support safe, efficient, and data-driven infrastructure management. However, balancing the need for detailed structural information with the practical challenges of sensor deployment and data management remains an important research challenge. This chapter introduces the context of the study and presents the background, motivation, aim, objectives, scope, limitations, contributions to knowledge, and the thesis structure.

1.1 Background

Bridge infrastructure forms the backbone of modern transportation systems, supporting urban mobility, economic activity, and resilient operations within smart cities and communities. However, these structures are subject to constant environmental and operational stresses that gradually degrade their material properties and can lead to catastrophic failures if not properly monitored and maintained (Sonbul & Rashid, 2023). Consequently, Structural Health Monitoring (SHM) has become an important component of intelligent infrastructure management. SHM is the process of implementing a damage identification strategy for aerospace, civil, and mechanical engineering infrastructure (Farrar & Worden, 2007). This process may include modal testing, signal processing, probabilistic analysis, numerical modeling, damage detection, damage diagnosis, damage reconstruction, and damage prediction (Mammeri et al., 2025). The traditional method of SHM relies on periodic visual assessments by trained engineers, but it is labour-intensive, time-consuming, and often fails to detect incipient damage before it becomes critical (Deng et al., 2023).

Recent advances in smart city technologies, the Internet of Things (IoT), and data-driven infrastructure systems have transformed bridge monitoring from periodic inspection to continuous and intelligent sensing. Modern bridge monitoring systems can now utilize sensor networks and real-time analytics to detect abnormal structural responses and provide early warning of potential problems (Sonbul & Rashid, 2023).

Among the structural responses commonly monitored are tilt and deflection, which are important indicators of structural behaviour under traffic loads, thermal actions, environmental effects, construction imperfections, and possible structural damage. Since these responses vary spatially along a bridge, measurements are typically required at multiple locations across the structure.

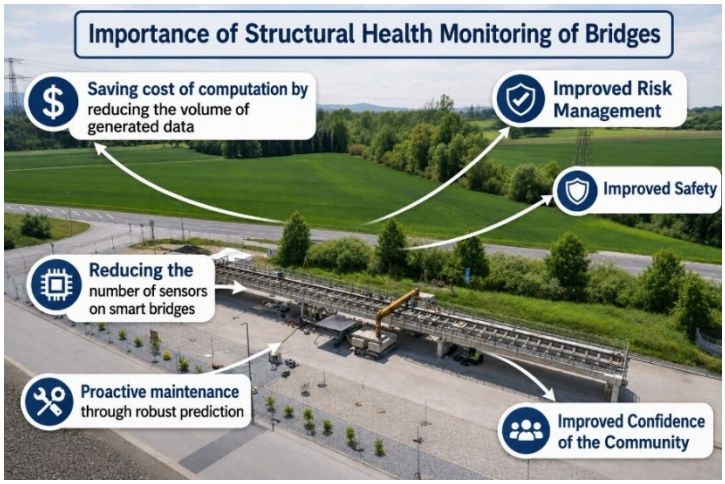


Figure 1: Importance of Structural Health Monitoring

Although, advanced sensing technologies such as distributed fiber-optic sensors (DFOS) can provide high spatial-resolution measurements along bridge structures, they generate large volumes of data that require substantial computational resources for transmission, storage, and real-time processing. Such large-scale deployment of dense sensing systems across urban infrastructure networks may be economically impractical for many smart-city applications. In contrast, discrete sensors (such as tiltmeters) can only take measurements at the exact location they are installed, leaving the structural response at every other location to be inferred. This creates a significant challenge for scalable and cost-effective bridge monitoring within smart infrastructure systems.

1.2 Motivation

The motivation for this thesis arises from a fundamental recognition that the challenges of “over-generated data” and “insufficient data” by the deployed sensors of smart city infrastructure monitoring are not insurmountable through technological advancement alone but rather require a paradigm shift in how we approach structural health

monitoring. Rather than deploying increasingly dense sensor networks to measure structural response, an alternative strategy is to deploy optimally located sparse sensors and use advanced physics-based computational techniques to infer the complete spatial distributions of structural responses from these limited measurements.

1.3 Aim and Objectives

The overarching aim of this thesis is to explore a Physics-Constrained Neural Network framework to predict and reconstruct the longitudinal tilt and deflection of a real-world bridge at every critical position where there are no sensors, as an attempt to close the gap between “over-generated data” from high spatial-resolution sensors and “inadequate data provision” from discrete sensors. To achieve this aim, the research is organized around the following specific objectives:

- **Obj 1** Identify the structural properties, material characteristics, and loading phases of the IDA-KI OpenLab bridge from the in-house documentation.
- **Obj 2** Develop an Autoregressive ML model to predict days-ahead, the signals of sensors.
- **Obj 3** Formulate the governing equations for the longitudinal tilt and vertical deflection.
- **Obj 4** Reconstruct the full-field tilt and deflection with PCNN under known physics conditions.
- **Obj 5** Develop an FEM numerical baseline model by reverse-engineering the net load and forward-engineering the spatial tilt/deflection fields using beam theory and known boundary conditions.

These objectives translate into the following research questions that will guide the investigation:

- **RQ1:** How can a physics-constrained neural network be configured to accurately reconstruct complete spatial distributions of bridge tilt from discrete sensors?

- **RQ2:** What is the minimum number of sensors required to achieve a specified reconstruction accuracy, and how does optimal sensor placement affect performance?
- **RQ3:** How do the reconstructions from the PCNNs compare in accuracy and computational efficiency to alternative methods such as FEM?

1.4 Research Scope

The analysis is limited to longitudinal tilt and vertical deflection of a two-span girder. While bridges experience multiple types of deformation, including lateral tilt, cracks, vibration, and foundation settlement, this thesis focuses exclusively on longitudinal tilt and vertical deflection.

1.5 Limitation

The validation of the Physics-Constrained Neural Network framework was initially intended to be performed using measurements from a held-out sensor on the bridge. However, the held-out accelerometer was damaged and unavailable during the study. Consequently, the reconstructed tilt and deflection responses were validated against a Finite Element Method (FEM) model instead of independent field measurements.

1.6 Contribution to Knowledge

Existing studies on inverse problems in structural health monitoring (SHM) of bridges typically rely on multiple sensors to reconstruct structural responses at uninstrumented locations. However, the hybrid AR-PCNN model utilizes the measurements from only one sensor per span of a continuous girder. To the best of the author's knowledge, no prior study has applied a physics-constrained neural network to achieve full-field reconstruction of both slope and deflection for a continuous girder under such sparse sensing conditions. Additionally, AR-PCNN is a predictive and reconstructive model, in which its predictive capability can be implemented for sensor anomaly detection by

flagging discrepancies between the forecasted and measured signals when they exceed a defined threshold.

1.7 Structure of Thesis

This thesis is organized into seven chapters, structured as follows:

- Chapter 1 establishes the background and motivation of the thesis, the specific aims and objectives, research scope, limitations, and knowledge contributions of the work.
- Chapter 2 reviews literature on the previous works across the domain of structural health monitoring systems, bridge responses, and physics-based computations.
- Chapter 3 introduces the case study of the IDA-KI bridge. Discussions on the measured quantities, the wind load data, and the boundary conditions of the bridge.
- Chapter 4 establishes the data preprocessing analysis, including data cleaning, feature transformation, correlation analysis, and information theory.
- Chapter 5 details the mathematical framework of the study, including the governing differential equations for tilt and deflection; the design of the architecture of the interconnected models; and the training and optimization procedures.
- Chapter 6 presents the empirical results from the models in Chapter 5 with detailed applicable discussions.
- Chapter 7 summarizes the key findings and contributions and concludes with reflections on the implications for smart bridge monitoring and infrastructure management in smart cities.

2 Literature Review

Several bridges across the world have experienced catastrophic failure associated with uncontrolled tilting, manifesting as a consequence of underlying mechanisms such as foundation scour, structural degradation, or impact-induced misalignment (Faulkner et al., 2020; Millar et al., 2024). Between 1980 and 2012, a total of 1062 bridge failures were reported in the United States, causing huge losses to the nation (Lee et al., 2013), many of which could have been avoided with the implementation of a structural health monitoring system. Bridge failures not only cause casualties and loss of lives, but they also result in economic crisis as a result of services that would need to come to a halt (Cook et al., 2015; Diaz et al., 2009; SMITH, 1976).

According to Lee et al. (2013) and Wardhana & Hadipriono (2003), bridge failures can be categorized into three: Distress, partial collapse, and total collapse. Distress is a failure that doesn't necessarily affect the usability of the bridge, but it may or may not result in a collapse (Lee et al., 2013; Wardhana & Hadipriono, 2003). Examples of such failure are deflection and tilt, which are the focus of this thesis.

2.1 Bridge Response

When a bridge is subjected to mechanical or environmental loading, they give responses in the form of deflection, tilt, crack, vibration, expansion, twisting, etc. These responses are then interpreted by structural engineers to ascertain if they are within the allowable limits or not. If one or a combination of these responses exceeds the allowable limits, there is a risk of partial or total collapse.

2.1.1 Tilt

In simple terms, tilt is the change in the angle of a structure relative to a perfectly level horizontal or vertical line (the gravity vector)(Charles & Skinner, 2004). Traditionally, tiltmeters and inclinometers (sometimes used interchangeably) are used to measure the angular tilt or rotation of a structural member because of their sensitivity for tracking

very small and precise measurements. Tiltmeters work by measuring the responses of the “pendulum behaviours” caused by gravity (Ha et al., 2013). A biaxial tiltmeter measures tilt in two distinct axes - the longitudinal and transverse axes. For each axis, the gravity vector is projected on the axis of the acceleration. The Sisgeo 0S542HD0502 D-tiltmeter used for this thesis uses a MEMS technology with an enclosed design to ensure high accuracy and low dependence on thermal factors and dynamic movements.

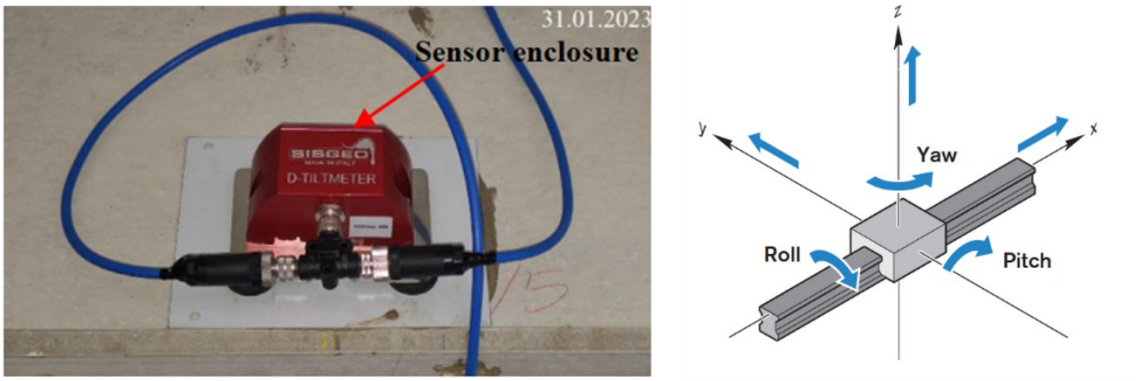


Figure 2: (a) Sisgeo 0S542HD0502 D-tiltmeter (b) Degree of freedom in space

Sources: (Jansen et al., 2025) and (Danielle Collins, n.d.)

The girder of a bridge can tilt about the X, Y, and Z directions. The rotation of the girder about the X-direction is called “roll”, about the Y-direction is called “pitch”, and about the Z-direction is called “yaw”. The kind of tilt measured by the Sisgeo 0S542HD0502 is pitch and roll (Srl, n.d.).

Pitch is primarily caused by the bending moment and flexural stress (Amaral & Mazzilli, 2017), emanating from self-weight, live loads from vehicular traffic, temperature variations, creep, and shrinkage – all of which self-weight is the dominant forcing (Mozumder et al., 2026). This form of tilt is also called slope (symbolized as θ), and the unit could be mm/m, radians, or degrees. The scope of this thesis is limited to pitch. Hence, roll and yaw will not be discussed further.

2.1.2 Deflection

Deflection is the temporary bending or displacement of a structural member (like a girder) from its original position when subjected to an external load, force, or gravity (Tang et al., 2024). Unlike pitch, which can simply be monitored with a tiltmeter/inclinometer, the monitoring of deflection is quite cumbersome. Traditionally, deflection can be monitored directly with Linear Variable Displacement Transducers (LVDTs), Laser Deflectometers, or Radar Interferometers. However, all these techniques require a reference position for the measurement because the sensors are not directly installed on the bridge. Because of this reference position, deflection measurement with the aforementioned techniques becomes impractical, especially when the bridge is constructed over a lake, river, muddy regions, or a major highway, where a reference point cannot simply be made (Hou et al., 2005; Sanli et al., 2000).

Many researchers have postulated that rotation measurement sensors (tiltmeters/inclometers that measure pitch) can be indirectly used to measure deflection of a bridge. We can infer deflection from pitch since there is a mathematical relationship between them. Hence, this creates an opportunity to monitor both tilt and deflection with a single sensor.

$$\theta = \frac{d\delta}{dx} \qquad \textbf{(eqn 1)}$$

Where δ is deflection and θ is pitch (might be referred to as slope or tilt in this thesis)

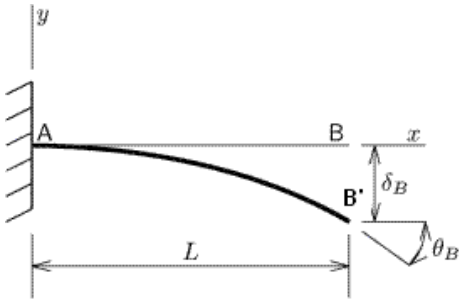


Figure 3: Deflection Diagram of a Cantilever Beam

Figure 3 shows the deflection diagram of a cantilever beam that is fixed supported at A and free at B. The line AB represents the original shape of the beam, and with the effect of external forcing, the deflected shape AB’ is obtained. The deflection at point B can be

computed as the vertical distance between B and B', while the slope at B (θB) is simply the angle between the tangent to the deflected curve at point B' and the original beam axis AB.

2.2 Forms of Analysis

For spatio-temporal analysis of bridge behavior, structural engineers employ three primary forms of analysis, namely, deterministic methods, data-driven approaches, and physics-based methods. Each approach addresses specific challenges in real-time monitoring, prediction accuracy, and computational efficiency.

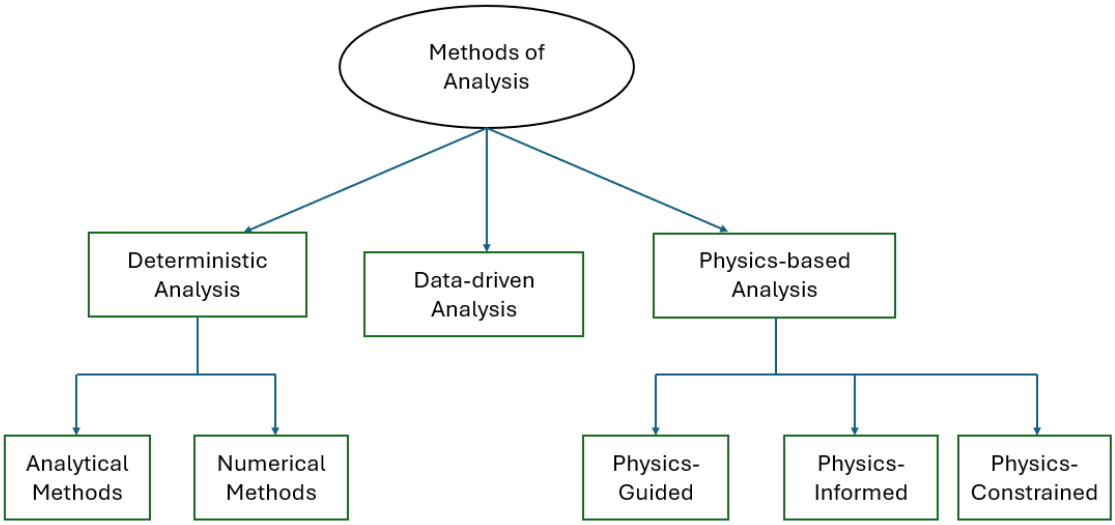


Figure 4: Forms of Analysis in Structural Health Monitoring

The selection of an analysis method depends on factors such as data availability, required accuracy, computational resources, and interpretability requirements. Figure 4 shows the breakdown of the analysis methods.

2.2.1 Data-Driven Analysis

Data-driven analysis relies primarily on learning patterns and relationships directly from observed data without explicitly enforcing the governing physical laws of the system. In a standard Feedforward Neural Network (FFN), inputs are passed through interconnected layers, where the network learns by adjusting weights and biases to minimize a

loss function through backpropagation and gradient-based optimization. According to the Universal Approximation Theorem, a sufficiently large neural network can approximate any continuous nonlinear function:

$$f(x) \approx \hat{f}(x;\theta) \quad \textbf{(eqn 2)}$$

where $f(x)$is the target function and $\hat{f}(x;\theta)$ is the neural network approximation parameterized by θ.

This capability has made FFNs widely applicable in prediction and pattern recognition problems. However, purely data-driven models become problematic for engineering systems governed by physical laws because a data-driven model can fit available data while violating the underlying physics, especially when data are sparse or noisy. In real-world SHM applications, where complete field measurements are often unavailable, relying solely on data-driven analysis may therefore produce physically unrealistic predictions despite achieving low numerical error.

2.2.1.1 Autoregressive (AR) Model

The Auto-Regressive (AR) model is a classical data-driven time-series forecasting method that predicts future values using previously observed measurements. Unlike deep learning models, AR models do not rely on neurons or hidden layers. Instead, they model temporal relationships directly through statistical dependencies between past and present observations. Due to their simplicity and relatively small number of trainable parameters, AR models are particularly effective for small datasets. To reduce overfitting in AR, regularization techniques are commonly introduced into the learning algorithm. Two widely used approaches are Ridge and Lasso regularization. Ridge regression penalizes the squared magnitude of model coefficients, preventing excessively large coefficients:

$$L = \sum_{i=1}^{N} (y_i - \hat{y}_i)^2 + \lambda \sum_{j=1}^{p} a_j^2 \quad \textbf{(eqn 3)}$$

while Lasso regression penalizes the absolute magnitude of the coefficients:

$$L = \sum_{i=1}^{N} (y_i - \hat{y}_i)^2 + \lambda \sum_{j=1}^{p} |a_j| \quad \textbf{(eqn 4)}$$

λ is the regularization coefficient that controls the penalty.

2.2.1.2 Neural Networks

Unlike traditional machine learning models that often rely on predefined features or simpler statistical relationships, neural networks automatically learn hierarchical representations through interconnected layers of neurons. A typical neural network architecture consists of an input layer, hidden layers, and an output layer. Each layer is composed of neurons that receive weighted inputs, apply an activation function, and pass the result to the next layer. The output of a neuron is given by:

$$y = \sigma(\sum_{i=1}^{n} w_i\, x_i + b) \qquad \textbf{(eqn 5)}$$

where x_i are the inputs, $w_i$are the weights, $b$is the bias term, and $\sigma(\cdot)$is the activation function. During training, the network adjusts its parameters to minimize prediction error using backpropagation and gradient-based optimization. Depending on the network architecture, neural networks may contain hundreds, thousands, or even millions of trainable parameters and therefore require substantial amounts of data to generalize effectively and avoid overfitting.

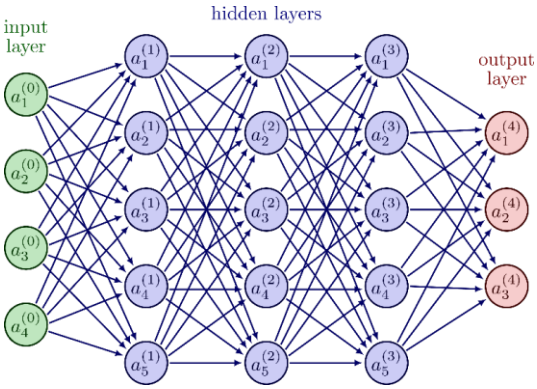


Figure 5: Neural Network Architecture

Source: Neutelings, I. (2021). Neural networks. Retrieved from https://tikz.net/neural_networks/

When datasets are limited, simpler machine learning models such as Auto-Regressive (AR) models may outperform deep learning architectures such as Long Short-Term Memory (LSTM) or Gated Recurrent Unit (GRU) networks, whose large number of

parameters can cause them to memorize training data rather than learn the true underlying temporal behaviour.

2.2.1.3 Fourier Cyclic Encoding

Fourier cyclic encoding is a feature engineering technique used to represent periodic behaviour in time-series data. Instead of using raw numerical values directly, cyclic variables are transformed into sine and cosine components so that their periodic relationships are preserved:

$$x_{sin} = \sin(2\pi f x) \; and \; x_{cos} = \cos(2\pi f x) \qquad \textbf{(eqn 6)}$$

where x is the variable and f is its frequency. This means that a single feature can be expanded into multiple sine and cosine features at different frequencies.

In bridge monitoring, cyclic encoding is useful because many structural responses follow periodic trends. Time-related features may contain daily or seasonal patterns, spatial coordinates may exhibit repeating structural behaviour, and environmental variables such as temperature may introduce cyclic thermal effects on the bridge response. Fourier cyclic encoding helps the model to learn periodic relationships that may not be obvious from the raw data alone.

2.2.2 Finite Element Method

The Finite Element Method (FEM) is a numerical analysis technique used to approximate the behaviour of engineering structures by dividing them into smaller interconnected elements. In structural engineering, FEM is widely used to compute deflection, slope, strain, and stress responses under different loading conditions. The discretized structural equilibrium equation is generally written as:

$$[K]u = F \qquad \textbf{(eqn 7)}$$

where $[K]$ is the global stiffness matrix of the structure, $\{u\}$ is the nodal response vector, and $\{F\}$ is the external load vector. Once the stiffness matrix and boundary conditions are defined, the structural response can be solved numerically.

In a reverse-engineering problem, measured structural responses are used to estimate unknown loads, stiffness properties, or boundary conditions through FEM calibration. One important property that makes this possible is the principle of FEM linearity. For a linear elastic system, the structural response is proportional to the applied load. This means that if a unit load produces a unit response $\{u_1\}$, doubling the load will double the response as well. Similarly, the response from multiple loads can be obtained by superposing the individual responses:

$$\{u\} = q_1\{u_1\} + q_2\{u_2\} + \cdots + q_n\{u_n\} \quad \textbf{(eqn 8)}$$

Since the measured structural response can be represented as a linear combination of unit-load responses, unknown loads can be reconstructed by solving for the scaling coefficients q_i.

2.2.3 Physics – Informed Neural Network

Physics-Informed Neural Network (PINN) is a framework that integrates physical laws into the training of neural networks for solving forward and inverse problems governed by ordinary differential equations (ODEs) or partial differential equations (PDEs) (Raissi et al., 2018). Instead of modifying the neural network architecture, PINNs incorporate the governing physical equations into the loss function. The PINN governing equation to approximate an unknown solution $u(x,t)$ can be expressed as a nonlinear differential operator:

$$\mathcal{N}[u(x,t)] = 0 \quad \textbf{(eqn 9)}$$

where $\mathcal{N}$ represents the physics of the system. The neural network approximation $u_\theta(x,t)$is then substituted into this operator to form a residual:

$$r(x,t) = \mathcal{N}[u_\theta(x,t)] \quad \textbf{(eqn 10)}$$

This residual is evaluated at a set of collocation points sampled within the spatio-temporal domain. The residual is typically expressed using a mean squared error (MSE) formulation and is referred to as the physics loss:

$$L_{physics} = \frac{1}{N}\sum_{i=1}^{N}\|r(x_i, t_i)\|^2 \qquad \textbf{(eqn 11)}$$

This physics loss is then incorporated as a key component of the total loss function of the model. Additional loss terms can be introduced to enforce the boundary and initial conditions of the system. Furthermore, when measurement data are available, a data-driven loss term is included to penalize the discrepancy between the predicted and observed values. The overall loss function can therefore be expressed as:

$$\mathcal{L}_{total} = \lambda_{data} \cdot \mathcal{L}_{data} + \lambda_{PDE} \cdot \mathcal{L}_{PDE} + \lambda_{boundary} \cdot \mathcal{L}_{boundary} + \lambda_{initial} \cdot \mathcal{L}_{initial} \qquad \textbf{(eqn 12)}$$

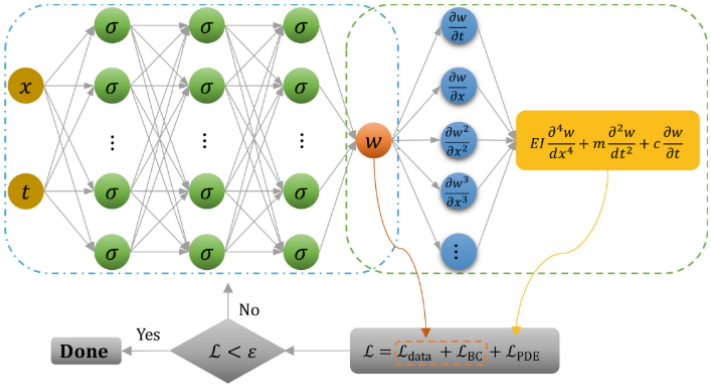


Figure 6: Physics-Informed Neural Network Architecture
Source: (Đạt et al., 2023)

The procedure for training a PINN model can be summarized as follows:

1. Define the governing differential equation(s) along with the domain of interest.
2. Specify boundary and/or initial conditions associated with the problem.
3. Construct a neural network approximation $u_\theta(x, t)$for the unknown solution.
4. Formulate the residual of the governing equation by substituting the neural network into the differential operator: $r(x, t) = \mathcal{N}[u_\theta(x, t)]$
5. Sample collocation points in the domain and compute the physics loss as the mean squared error of the residual.

6. Define additional loss terms for boundary conditions, initial conditions, and (if available) observational data.
7. Minimize the total loss function using gradient-based optimization (e.g., backpropagation), thereby updating the network parameters θ.

2.2.3.1 Collocation Points

Collocation points are selected discrete locations in space, time, or both, where a mathematical model is forced to satisfy a governing equation or physical condition (Bauduin et al., 2025). They are "checkpoints" used to test whether a numerical or machine learning solution obeys the physics of a problem. In a PINN solution, the residual of a differential equation is computed at the carefully selected collocated points. The idea is that if the solution behaves correctly at these points, then it will approximate the true physical behaviour of the entire system.

2.2.4 Physics – Constrained Neural Network

Some of the challenges associated with PINN have led to the development of Physics-Constrained Neural Network (PCNN). The challenges include:

Challenge 1: The summation of all the loss components in (eqn 12) results in an unstable and imbalanced magnitude of the gradient during backpropagation (Wang et al., 2020).

Challenge 2: PINN treats physical constraints purely as soft penalties added to the loss function. This ensures the model is informed of the physics but doesn't strictly ensure it respects the physics at all times.

Wang et al (2020) proposed an adaptive learning rate annealing algorithm that balances the magnitudes of different loss terms as an attempt to solve challenge 1. Challenge 2 birthed the advent of PCNN, where physical constraints are rather treated as a hard penalty (Geneva & Zabaras, 2019; Lu et al., 2021; Wang et al., 2020).

The advantages of PCNN over PINNs include:

1. Guaranteed physical consistency by construction, not by optimization

2. Simplified loss function.
3. Faster convergence during training.

PCNN enforces hard constraints via trial functions, Signed Distance Functions (SDF), Mixed Formulations, or Constraint Projections. Trial functions are best used for simple and regular domains (Lagaris et al., 1998). SDFs are most suitable for irregular and non-convex geometries (Kraus & Tatsis, 2024). Mixed formulations are best for higher-order PDEs (Rezaei et al., 2022), and Constraint projections are best for conservation law problems (Baez et al., 2024). This thesis utilizes a trial function approach because it has little computational overhead, and the geometry of our girder (T-section) is simple and regular throughout the spans.

2.2.4.1 Governing Equation

A governing equation is a mathematical expression of a theory that describes the physical behaviour of a system. In structural mechanics, governing equations are used to relate quantities such as load, displacement, stress, strain, slope, and vibration. One of the most widely used governing equations in bridge analysis is the Euler–Bernoulli beam equation. The theory assumes that plane sections remain plane after deformation and is commonly applied to slender beams undergoing small deflections. The dynamic form of the Euler–Bernoulli beam equation is given as:

$$EI\frac{\partial^4 w(x,t)}{\partial x^4} + m\frac{\partial^2 w(x,t)}{\partial t^2} = q(x,t) \qquad \textbf{(eqn 13)}$$

where E is the Young's modulus, I is the second moment of area, $w(x,t)$is the beam deflection, m is the mass per unit length, and $q(x,t)$ is the distributed load. The second term at LHS represents inertial effects associated with vibration and dynamic motion. For static problems where inertial effects are negligible, the equation reduces to:

$$EI\frac{d^4 w(x,t)}{dx^4} = q(x) \qquad \textbf{(eqn 14)}$$

In this study, the static form is adopted because the objective is to reconstruct quasi-static bridge responses dominated primarily by slowly varying environmental and operational loading rather than high-frequency dynamic vibration effects.

2.2.4.2 Nondimensionalization of Governing Equation

Non-dimensionalization is the process of removing units of measurement from a mathematical equation or physical model by scaling the variables against intrinsic or characteristic properties. PINNs perform badly on dimensional equations when coefficients span many orders of magnitude (Kapoor et al., 2024). The idea of nondimensionalizing PDEs for PINN application was first postulated by Kapoor et al. (2024) where the model's spatial coordinate x, time t, displacement u, and force f were nondimensionalized as:

$$\bar{x} = \xi_1(x); \qquad \bar{t} = \xi_2(t); \qquad \bar{u} = \xi_3(u); \qquad \bar{f} = \xi_4(f)$$

where ξ1, ξ2, ξ3, and ξ4 are suitable functions that map the dimensional quantities $\bar{x}, \bar{t}, \bar{u}, and\ \bar{f}$ to the corresponding nondimensional quantities and then substituted into the governing PDE.

Using the static Bernoulli equation, (eqn 14), in its original form could cause a gradient imbalance due to differences in feature magnitude. We could nondimensionalize the PDE itself by dividing (eqn 14) by EI. Hence, the nondimensionalized Euler-Bernoulli ODE becomes:

$$\frac{d^4w}{dx^4} + \frac{q}{EI} = 0 \qquad \textbf{(eqn 15)}$$

2.2.4.3 Boundary Conditions

Boundary conditions define the physical constraints imposed on a structure at its boundaries or supports. They describe how the structure is allowed to move or rotate under loading. For example, a fixed support prevents both displacement and rotation:

$$w(0) = 0, \qquad \theta(0) = 0$$

while a simply supported beam prevents displacement but allows rotation:

$$w(0) = 0$$

2.2.4.4 Trial Function

Trial functions, also called auxiliary functions or ansatz, are mathematical functions used to hard-code boundary conditions (BCs) directly into the solution of a neural network (Manavi et al., 2024). In the narrative of modern PINNs, trial functions move boundary conditions (BCs) from the loss function directly into the architecture of the solution. By "hard-coding" these constraints, the model ensures that no matter how the weights change during training, the boundaries remain mathematically perfect (Chung et al., 2026). For instance, in a standard PINN framework, the governing physics of a hidden solution *u(X,t)* can be defined by a general linear or nonlinear operator *F* within a domain *Ω*, subject to boundary conditions *B*[*u*](*X*,*t*)=0 on ∂Ω and initial conditions *u*(*X*,0)=*S*(*X*). The training will have to utilize a composite loss function of the form

$$Loss_{total} = Loss_f + Loss_b + Loss_i \quad \textbf{(eqn 16)}$$

This composite loss function seeks to minimize the residuals of the PDE ($Loss_f$), boundary ($Loss_b$), and initial conditions ($Loss_i$) simultaneously. However, this multi-objective optimization often suffers from unstable gradient pathologies and requires meticulous tuning of penalty weights to achieve convergence. To eliminate the sensitivities of soft enforcement, the structure of the solution can be transformed via a trial function (Manavi et al., 2024). A multiplicative trial solution can be given by:

$$u^*(X,t) = G(X,t) + D(X,t) \times \hat{u}(X,t,G,D) \quad \textbf{(eqn 17)}$$

Where $u^*(X,t)$ is the trial solution, G(X,t) is a pre-trained network representing the boundary/initial state, D(X,t) is a distance function that splits the boundary from the collocation points, and $\hat{u}(X,t,G,D)$ is a neural network with adjustable parameters (Manavi et al., 2024). Because the D(X,t) vanishes at the boundaries, the trial solution $u^*(X,t)$ is forced to equal $G(X,t)$ regardless of the parameters of the main network. Consequently, $Loss_b$ and $Loss_i$ become zero by design, allowing the total loss function, (eqn 16), to simplify into a single-objective PDE residual:

$$loss_{total} = loss_f(\theta) = \frac{1}{N_f}\sum_{i=1}^{N_f}\left\|\widehat{f}_\theta(X,t)\right\|^2 \quad \textbf{(eqn 18)}$$

Rather than having a pre-trained network and distance function to enforce boundary and/or initial conditions into the output of a NN, a simpler way to go about this is with a polynomial function. In the physical system of the girder analysed in this thesis, the support at A, B, and C is fixed, fixed, and pinned, respectively (refer to Figure 10 or Figure 13). We are required to enforce six boundary conditions, two per support:

$$w(0) = 0;\ \frac{dw}{dx}(0) = 0,\ w(L) = 0;\ \frac{dw}{dx}(L) = 0,\ w(2L) = 0;\ and\ \frac{d^2w}{dx^2}(2L) = 0$$

x is any position on the girder, including the collocation points, i.e.,

$$0 \le x \ge L \text{ in span AB and } L \le x \ge 2L \text{ in span BC.}$$

These boundary conditions can be enforced by hard constraints by expressing the deflection as the product of a polynomial trial function ϕ(x) and a neural network output:

$$w(x,t) = \varphi(\mathrm{x})\ \mathrm{x}\ \mathrm{NN}(\mathrm{x},\mathrm{t},\mathrm{T},\mathrm{phase}) \quad \textbf{(eqn 19)}$$

A typical polynomial trial function that will enforce the above boundary conditions can be given as follows:

$$\varphi(x) = \frac{x^2(x-15)^2(x-30)}{\varphi\_scale} \quad \textbf{(eqn 20)}$$

Where its first and second derivatives are:

$$\frac{d\varphi(x)}{dx} = \frac{5x(x-15)(x^2-33x+180)}{\varphi\_scale} \quad \textbf{(eqn 21)}$$

$$\frac{d^2\varphi(x)}{dx^2} = \frac{4x^3-144x^2+1350x-2700}{\varphi\_scale} \quad \textbf{(eqn 22)}$$

φ_scale is a scaling factor that normalizes the function. It can be expressed as a function of the collocation points:

$$\varphi\_scale = \max(|x_{col}^2(x_{col}-15)^2(x_{col}-30)|) \quad \textbf{(eqn 23)}$$

However, choosing a single polynomial trial function for the whole spatial domain becomes difficult for a continuous girder because of the difference in support types. An alternative solution might be to use a piecewise polynomial trial function across the spans.

2.3 Notable Reviewed Literature

The use of rotational measurements (tiltmeters/inclinometers) for full-field deflection estimation of bridges has gained increasing attention in Structural Health Monitoring due to the limitations of conventional displacement measurement techniques, particularly for bridges spanning rivers, highways, and inaccessible terrain. The underlying principle is based on classical beam theory, where beam tilt is related to the spatial derivative of deflection, as in (eqn 1)

A.K Sanli et al (2000) developed one of the earliest reconstruction frameworks for deflection from a tiltmeter that is based on cubic spline interpolation for representing the bridge elastic curve. The spline formulation enforced continuity and compatibility conditions across adjacent beam segments using the spline equation depicted in the first row of Table 1. Their study demonstrated that tiltmeter-based reconstruction could effectively replace conventional displacement transducers for inaccessible bridges. However, the methodology is fundamentally dependent on multiple distributed slope measurements to preserve spline continuity and reconstruction stability.

X. Hou et al (2005) also proposed an inclinometer-based bridge deflection reconstruction method using assumed shape functions and least-squares estimation. The bridge deflection field was represented with the equation in the second row of Table 1. Multiple inclinometers were placed along a simply supported and a continuous span to measure tilt. Although they found good accuracy in their reconstruction, the authors concluded that at least 5 inclinometers/tiltmeters are required per span to get good accuracy because their reconstruction depended on solving multiple unknown shape-function coefficients through least-squares estimation.

Similarly, H. Shenton (2015) implemented a long-term SHM system on the Indian River Inlet cable-stayed bridge using nine fiber-optic tiltmeters distributed along the bridge deck. Their reconstruction approach relied on integrating measured rotations to obtain deck deflections according to:

$$w(x) = \int \theta(x)\, dx \qquad \textbf{(eqn 24)}$$

The reconstructed deformation profiles were validated against a calibrated finite element model under controlled truck loading. Their findings confirmed the feasibility of tilt-based monitoring for full-scale bridges and further highlighted the critical role of sensor placement in reconstruction accuracy. In particular, the authors reported underestimation of peak deflections because maximum rotation regions were not directly instrumented.

P. Ni et al (2022) implemented a physics-informed convolutional neural network to reconstruct bridge displacement from sparse measurements, whereby a CNN with two parallel branches was designed to output the quasi-static and dynamic displacement components. They validated their model with a laboratory beam test that used one tiltmeter at a support and one accelerometer at midspan to train the network. Unlike other papers that used dense sensors, this paper utilized two sensors per span coupled with a physics-informed computation to reconstruct deflection.

More recently, A.I.F Al-Adly & Kripakaran (2025) introduced a PINN framework for virtual sensing in beams subjected to moving loads. They embedded the governing Euler-Bernoulli beam equation directly into the neural network training process:

$$EI\frac{\partial^4 u(x,t)}{\partial x^4} + \mu\frac{\partial^2 u(x,t)}{\partial t^2} = P\delta(x - vt) \qquad \textbf{(eqn 25)}$$

They started with a 1D PINN (input = spatial coordinate x) and extended it to 3D (inputs x, time t, and load magnitude) to capture the moving axle. Training data came from a real girder under known vehicle crossings. The PINN accurately predicted deflection, strain, and bending moment at locations without sensors. However, their study used a single-span girder under a single moving load and relied on existing monitoring data at many locations. It does not address multiple spans or fixed mid supports, nor the challenge of only one tilt input.

Table 1: Summary Table of Notable Reviewed Literature

Authors	Governing Principle	Sensor Count	Support
(Sanli et al., 2000)	Cubic spline fit $S(x) = \frac{M_{i-1}(a_i - x)^3}{6h_i} + \frac{M_i(x - a_{i-1})^3}{6h_i} + \left(y_{i-1} - \frac{M_{i-1}h_i^2}{6}\right)\frac{a_i - x}{h_i} + \left(y_i - \frac{M_i h_i^2}{6}\right)\frac{x - a_{i-1}}{h_i}$	13 tiltmeters + 2LVDT	Continuous
(Hou et al., 2005)	Beam shape function: $y(x,t) = A(x)\sum_{j=1}^{n-1} X_j(t)g_j(x)$	≥5	Continuous
(Shenton et al., 2015)	Tilt integration: $w(x) = \int \theta(x)\,dx$	9	Continuous
(Ni et al., 2022)	CNN-based PINN	1 tiltmeter + 1 accelerometer	Simply supported
(Al-Adly & Kripakaran, 2025)	Euler-Bernoulli-based PINN: $EI\frac{\partial^4 u(x,t)}{\partial x^4} + \mu\frac{\partial^2 u(x,t)}{\partial t^2} = P\delta(x - vt)$	Distributed response info	Simply supported

3 Case Study

This section describes in detail the kind of bridge that is monitored, the condition of the site, the sensors that take the measurements, the layout of the sensors, the measurements being taken, the frequency of the measurements being taken, explanations of the various quantities, and the schema of the data.

3.1 Site Description and Research Context

The Infrastructure Data Analysis with Artificial Intelligence (IDA-KI) OpenLab Research Bridge is a full-scale prestressed concrete bridge located on the campus of Technische Universität Dresden (TU Dresden), Dresden, Saxony, Germany. Specifically, the coordinates are 51º 12’04.56” N and 14º 24’23.78” E. The structure functions as a permanent experimental platform for SHM research. Unlike operational bridges, where access is restricted and loading histories are often uncertain, the OpenLab bridge provides a controlled and realistic environment in which instrumentation strategies, data acquisition systems, and data-driven structural assessment methods can be developed and validated under known boundary and loading conditions. The bridge is 45m long with three spans, each 15m long, and has a width of 4.5m. Each span is built with different construction methods to provide a wide range of research focus. The first and second spans are built with a similar construction method, only that the prestressed elements of span 2 are embedded with a network of distributed fiber optic sensors to facilitate the distributed measurement of physical changes such as strain and temperature (*OpenLAB — Institute of Concrete Structures — TU Dresden*, 2023).

The collected data of the bridge are in two stages:

1. *Reference Phase*: This is the period between February 1, 2024, and October 31, 2024. This phase is the period after completion of the construction, whereby the bridge is only under the influence of climatic conditions and traffic loading. To simulate the effects of traffic, a rail-guided load vehicle is driven over the bridge several times a month. Our study uses the data from this phase.

2. *Static and Dynamic Load Test Phase*: This is the period after the reference phase in which the bridge is put to static and dynamic test via hydraulic pressure and a load traverse. This phase also includes additional excitation of the structure with a directed exciter so that the dynamic structural behavior can be investigated in different frequency ranges (*OpenLAB — Institute of Concrete Structures — TU Dresden*, 2023)

Figure 7: IDA-KI OpenLab Bridge During Load Test Phase.
Source: (Herbers et al., 2024; Richter et al., 2024)

3.2 Structural Geometry and Material Properties

As illustrated above, the bridge is composed of three spans. However, the third span is left out of the scope of this thesis due to the difference in its construction method, the prestressed element (PE) configuration, and the sensor installations on the bridge. There are three girders, each in spans AB and BC, of which the cross section is a T-beam. The single rail track is centralized on the deck. The cross-section of the girder and deck is shown in Figure 8.

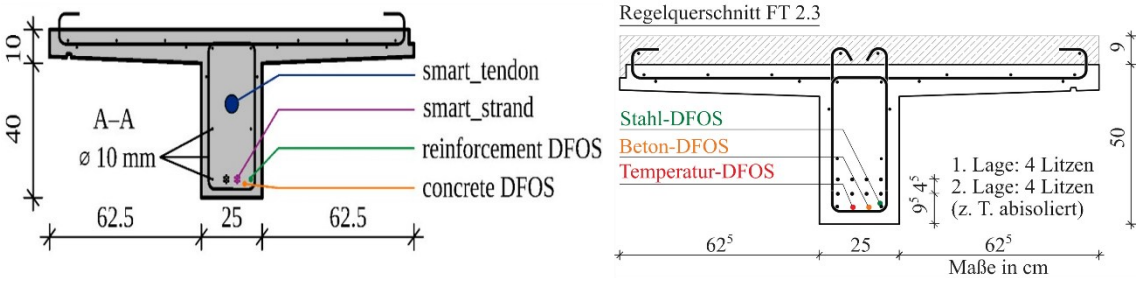


Figure 8: Cross-section of the girders and deck
Source: (Herbers et al., 2024; Richter et al., 2024)

Table 2: Material Properties of the mid girder

Property	Value	Description
Young's modulus, E	37.0 GPa	C40/50 concrete
Moment of Inertia, I	5.208×10^{-3} m^4	I_{zz}
Section Area, A_{girder}	0.25 m^2	Area of the T-section girder
Concrete Unit weight, γ_{PC} & γ_{RC}	25kNm^{-3}	Reinforced/Prestressed concrete
Deck thickness, t_{deck}	0.1m	-
Deck/Bridge width, b_{deck}	4.5m	-
Span length, L	15.0 m	Each span

3.3 Support and Boundary Conditions

The static system of the bridge evolved through several stages during erection.

I. *Stage 1 - Assembling of the prestressing elements (PEs)*: The system behaved as a series of simply supported girders with a short cantilever segment. As shown in Figure 10(a).

II. *Stage 2 - Casting of the in-situ concrete deck*: As shown in Figure 10(b), the system gradually transitioned into a frame-like system at this stage. The prestressing elements in axes 10 and 20 became monolithically connected to the substructures, while in axis 30.

III. *Stage 3 - Removal of the temporary scaffolding*: This stage represents the final static system of the bridge as shown in Figure 10(c). Owing to the significantly higher stiffness of the abutment wall at axes 10 and 20, a fixed support condition can reasonably be assumed at both axes.

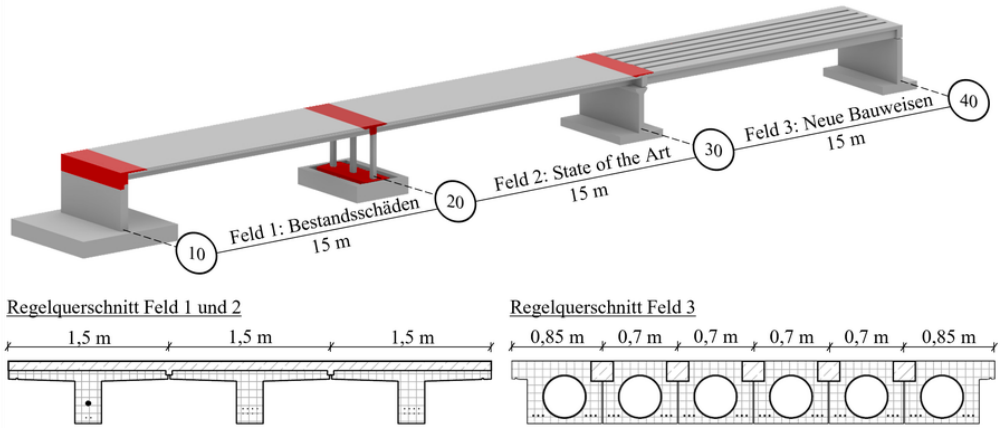


Figure 9: Movable and Non-movable Joints on the Bridge
Source: (*OpenLAB — Institute of Concrete Structures — TU Dresden*, 2023)

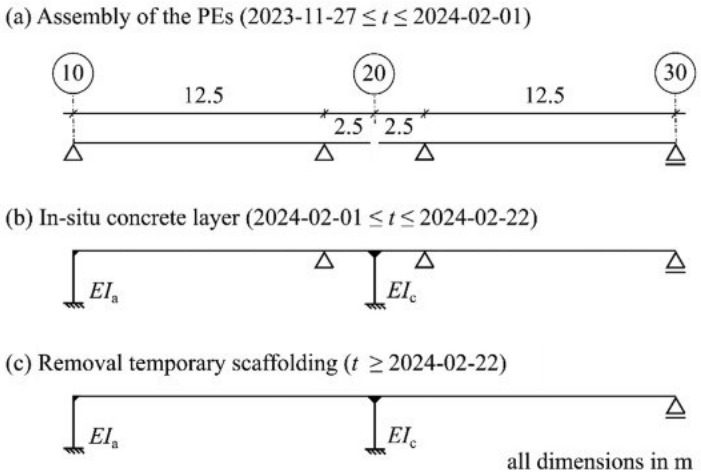


Figure 10: Evolution of the Static System of Span AB and BC
Source: (Jansen et al., 2025)

3.4 Sensor Layout

One Sisgeo OS542HD0502 digital tiltmeter is installed on each girder – 6 in total. Each tiltmeter is positioned 4 m from axis 20 (i.e., at x=11m and x=19m), approximately at the location of zero bending moment under self-weight, where the highest tilt is expected for Span AB. This is optimal for Span AB but not Span BC.

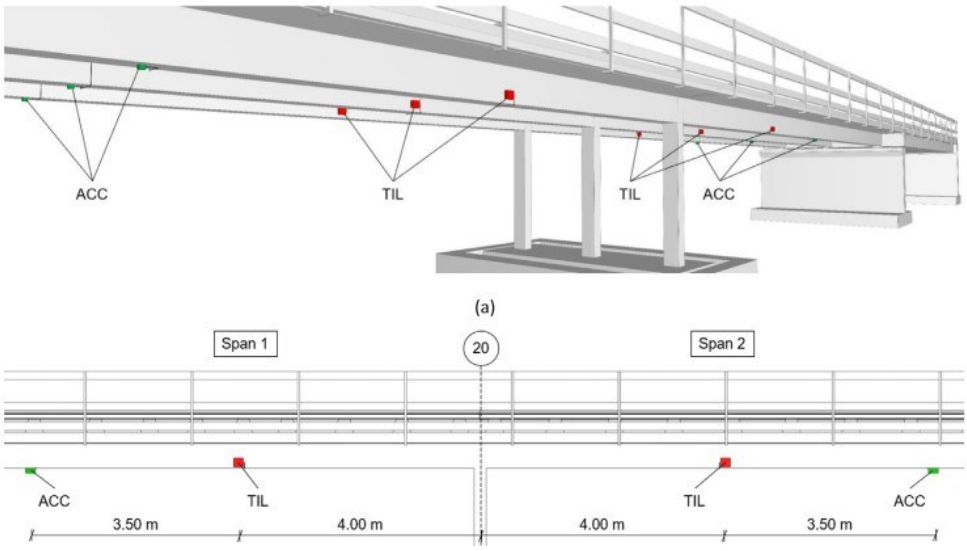


Figure 11: Sensor Layout for Tiltmeters (TIL) and Accelerometers (ACC)
Source: (Jansen et al., 2025)

3.5 Loading System

The bridge carries a single railway track made up of steel rails supported by prestressed concrete sleepers. When the test vehicle passes over the bridge, the load is first transferred to the rails and then distributed through the sleepers at equidistance onto the girders. These girders subsequently carry and distribute the load through the bridge structure. For this study, the primary interest is the load transmitted to the girder, since this directly influences the structural response measured by the tiltmeters.

Figure 12: Sleepers, Rail-tracks, and the Test Vehicle (Jansen et al., 2025)

The bridge loading process was carried out sequentially throughout 2024. The girders were installed on February 22, followed by the railway tracks on March 18. On April 22, a 4.1-ton test vehicle load was introduced, and by June 28, the applied vehicle load had increased to 9.0 tons.

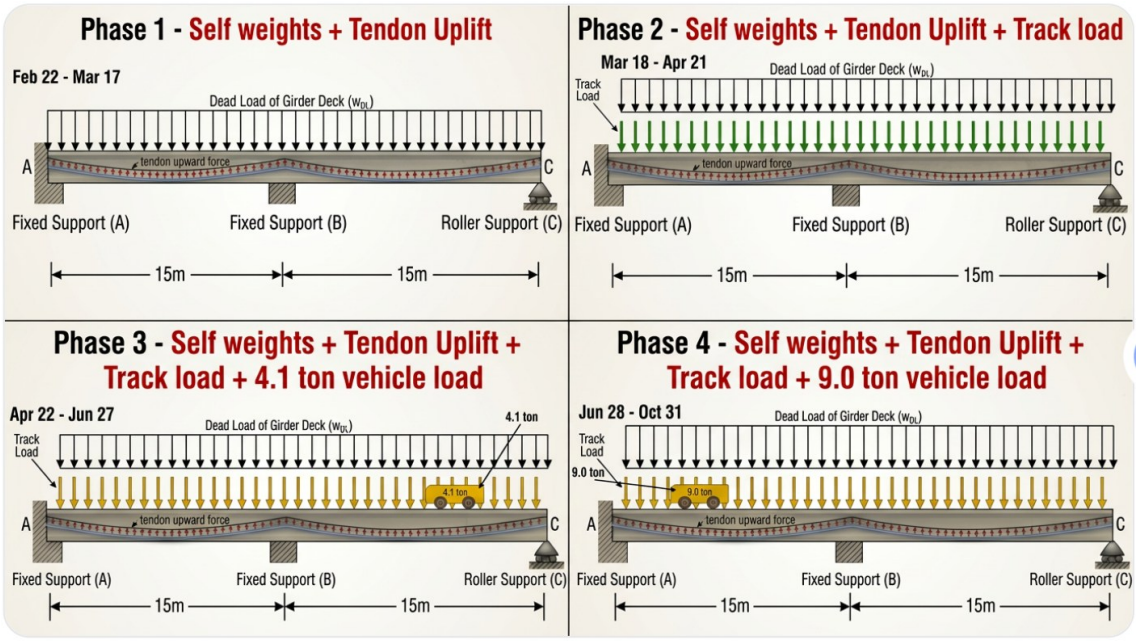


Figure 13: Loading Phases of the Girder

3.6 Measured Data

Each installed tiltmeter simultaneously provides, in real-time, the tilt along the transverse and longitudinal direction of the bridge relative to the installed position in mm/m, the surface temperature in °C, and the relative humidity in % at a sample rate of 1/600Hz i.e 10mins.

All measurements were obtained directly at the test site except for the wind data. Since wind speed and direction are important for analyzing bridge tilt behaviour, secondary data were obtained from the nearest weather station due to the absence of on-site wind measurements. Wind speed and direction measured at 10 m above ground level were obtained from the Deutscher Wetterdienst (DWD) Climate Data Center from Station 15911, located approximately 5 km from the test site. The data were sampled at the same frequency as the bridge measurements and had undergone DWD quality control procedures.

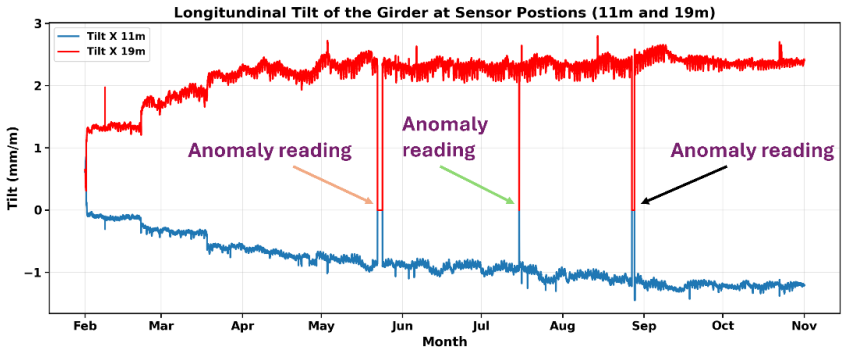
4 Data Preprocessing

This chapter presents the walkthrough of how the raw data is transformed into clean feature-rich data suitable for physics-constrained and autoregression modelling. This stage is critical because the performance of the models depends not only on the architecture but also on the fidelity and representativeness of the input data.

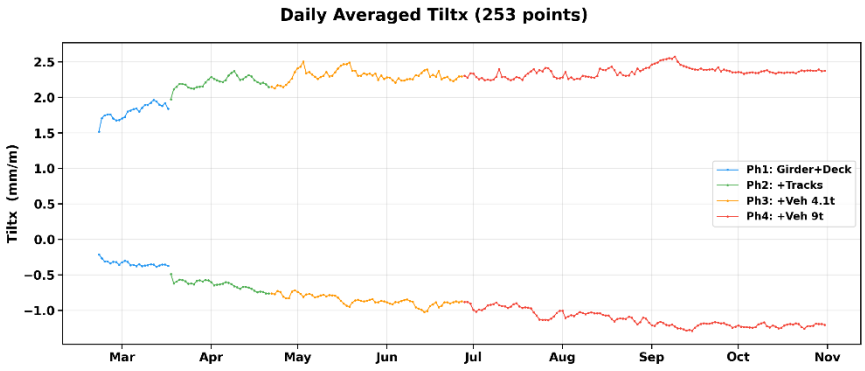
4.1 Data Cleaning

As shown in panel (a) of
Figure **14**, the measured raw data includes several spikes that possibly result from a power outage. Several duplicate samples are also present in the data. The duplicate samples are simply dropped, while the missing samples and the anomaly spikes are being imputed using linear interpolation, given by:

$$y_t = \left| \frac{y_{t-1} + y_{t+1}}{2} \right| \qquad \textbf{(eqn 26)}$$



(a) Raw data with spikes and duplicates



(b) Daily averaged data

Figure 14: Daily Averaging

It is important to resample the noisy 10-minute raw data to identify the data trend. Since the measurements are temperature-dependent, the ideal resampling method is to average the data by 24 hours to filter out the diurnal noise by approximately $\sqrt{144}$ and to standardize the data. The resampled data is shown in panel (b) of Figure **14**.

4.2 Linear Relationship

Correlation analysis helps us establish linear and/or monotonic relationships among variables. Pearson correlation coefficient (r) is used to measure the strength and direction of the linear relationship between two variables (features and/or target) and represents that relationship as a value between -1.0 and 1.0. The closer r is to |-1|, the stronger the strength. Pearson correlation coefficient is given by:

$$r = \frac{n\sum xy - (\sum x)(\sum y)}{\sqrt{[n\sum x^2 - (\sum x)^2][n\sum y^2 - (\sum y)^2]}} \qquad \textbf{(eqn 27)}$$

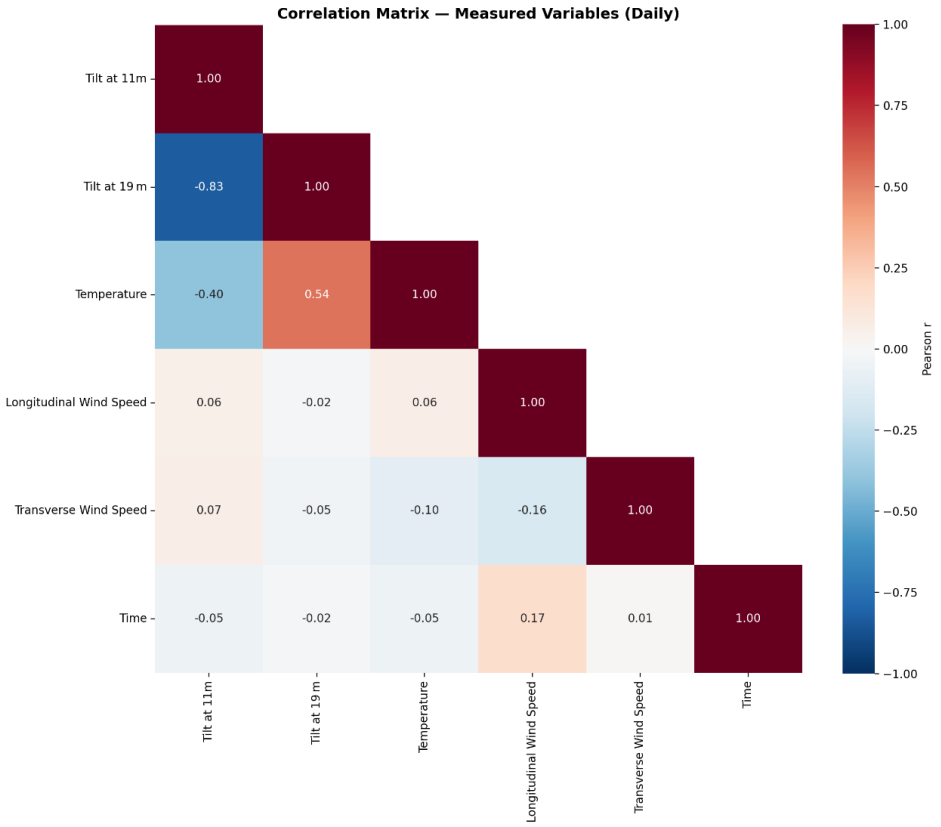


Figure 15: Correlation Matrix

It can be interpreted from

Figure **15** that there is a moderate linear relationship between the daily tilt and the daily temperature. The temperature-tilt correlation is negative in span AB and positive in span BC, suggesting that temperature variations are associated with opposite tilt trends in the two spans. In contrast, wind speed shows a weak linear relationship with the measured tilt. Wind primarily acts as a lateral load, but the measured tilt corresponds to the rotation of the girder about a horizontal axis that lies in the plane of the girder and perpendicular to the direction of traffic. The forcing directions do not coincide; hence, both features do not influence each other.

4.3 Feature Engineering and Transformation

The only available measured variables that can be used to derive information about tilt are temperature, wind speed, and time. However, as shown in

Figure **15**, wind speed and time exhibit very weak linear relationships with tilt, leaving temperature as the dominant variable. To extract additional information, new features are engineered from time and temperature through Fourier cyclic transformations, while wind speed is ignored in the analysis. The engineered features are:

A. Temperature-based features

- Standardized temperature: $T_{norm} = \frac{T-\mu_T}{\sigma_T}$
- Fourier-based features: Transforms T_{norm} to sine and cos features as per (eqn 6)
- Mean Temperature: $dT_x = \frac{\text{Temp. at the front of girder + Temp. at the back of girder}}{2}$
- Temperature Change: ΔT= Temp. at the front of girder − Temp. at the back

B. Time-based Fourier features (cyclic encoding)

- Normalized time: $t_{norm} = \frac{\text{day of the year}}{\text{total days in the dataset}-1}$
- Fourier-based features: Transforms t_{norm} to sine and cos features as per (eqn 6)

Where frequency f = [1,2,4,8,16], μ_T is the mean temperature, and σ_T is the standard deviation of temperature. Temperature is being standardised to ensure the Fourier-

based features operate on a comparable scale to the temporal features, preventing the neural network from being dominated by one input scale over another.

Table 3: Frequency Breakdown

Frequency, f	**1**	**2**	**4**	**8**	**16**
What it captures	Full seasonal cycle	Half-year variation	Bi-monthly patterns	Monthly fluctuations	Bi-weekly variation

Figure 16 shows the periodic progression of the engineered features at two frequencies: f = [4,16]

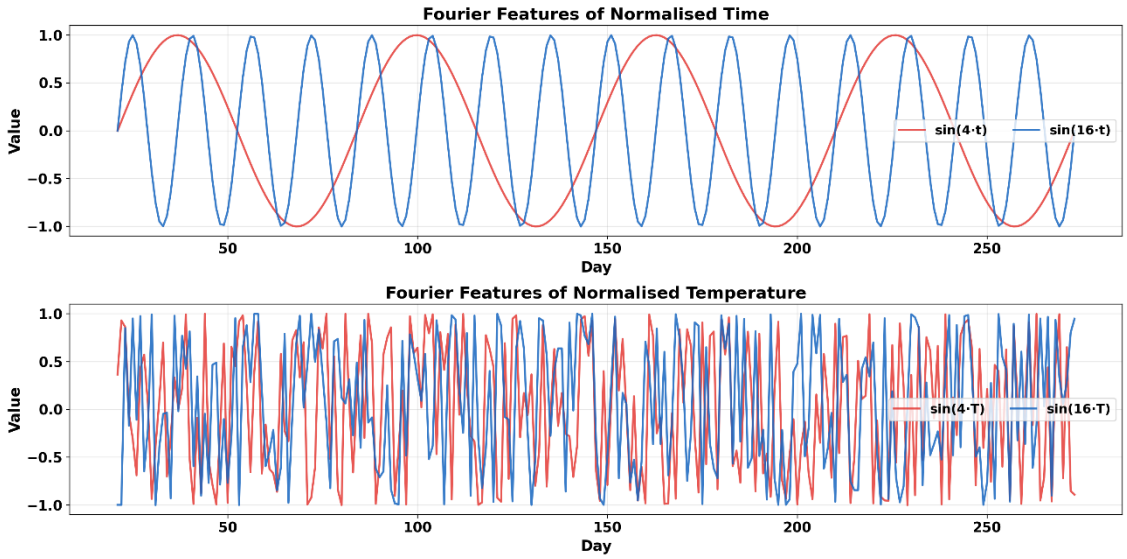


Figure 16: Fourier-based Features for Frequency 4 & 16

4.4 Mutual Information

Unlike Pearson correlation, which only captures linear relationships, mutual information (MI) captures any statistical dependency, including nonlinear ones. We compute MI between each feature and both tiltx targets to identify features with nonlinear predictive value that linear correlation might miss.

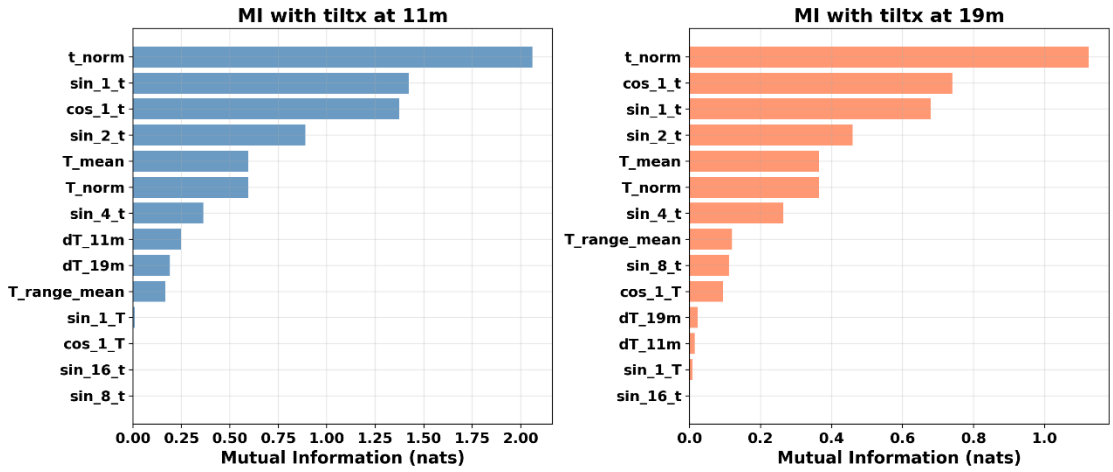


Figure 17: Mutual Information on the Measured Feature

The MI analysis confirms time as the dominant predictor. MI of t_norm is high because the time index encodes both the loading phase (step changes) and the seasonal temperature trend. However, we do not use t_norm directly; instead, we use its Fourier encoding, which provides the same information in a form that the neural network can process without spectral bias.

The Fourier features sin_1_t and cos_1_t (one full seasonal cycle) also carry substantial information. Interestingly, the higher-frequency Fourier features (f=8, 16) have lower MI, suggesting that most of the tilt variation is explained by slow (seasonal) trends rather than rapid oscillations. However, these high-frequency features may still be important for the PCNN's ability to capture day-to-day fluctuations within each phase.

4.5 Transfer Entropy

Transfer entropy measures directional information flow. It answers, “Does knowing past values of X help predict future values of Y better than knowing past Y alone?” This tells us whether temperature drives tilt (or vice versa), which has implications for the causal structure of our model. We approximate it as:

$$TE(T \rightarrow tilt) = MI\left(T_t, tilt_{t+lag}\right) - MI\left(tilt_t, tilt_{t+lag}\right) \quad \textbf{(eqn 28)}$$

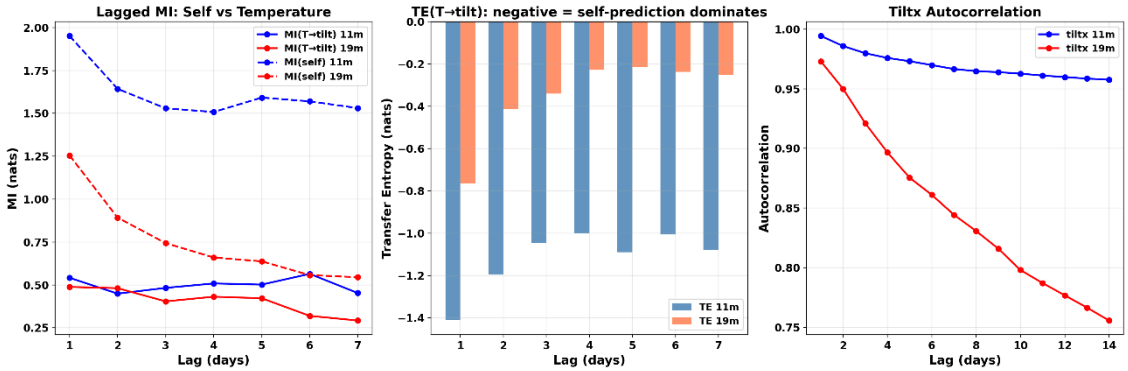


Figure 18: Transfer Entropy

Because transfer entropy is negative at all lags for both sensors, we can interpret it that tilt's own past is always more informative about its future than temperature is. Similarly, Self-MI decays slowly with lag, reflecting the high autocorrelation of the tilt signal.

5 Modelling

The central question of this thesis is twofold:

1. Can we predict future full-field responses before measurements are taken?
2. Can we reconstruct the full-field slope and deflection along the entire 30 m beam from only two tiltmeter measurements?

We address these through four interconnected models:

A. *AR Predictive Model*: an autoregressive model that forecasts the next day tiltx from historical measurements and weather data

B. *PCNN*: a physics-constrained neural network that reconstructs full-field slope and deflection from measured tiltx.

C. *AR-PCNN*: a hybrid architecture coupling the AR model with the PCNN for end-to-end prediction and reconstruction.

D. *FEM Baseline*: a deterministic finite element model that serves as the validation target (in place of actual measured responses, which do not exist on this bridge)

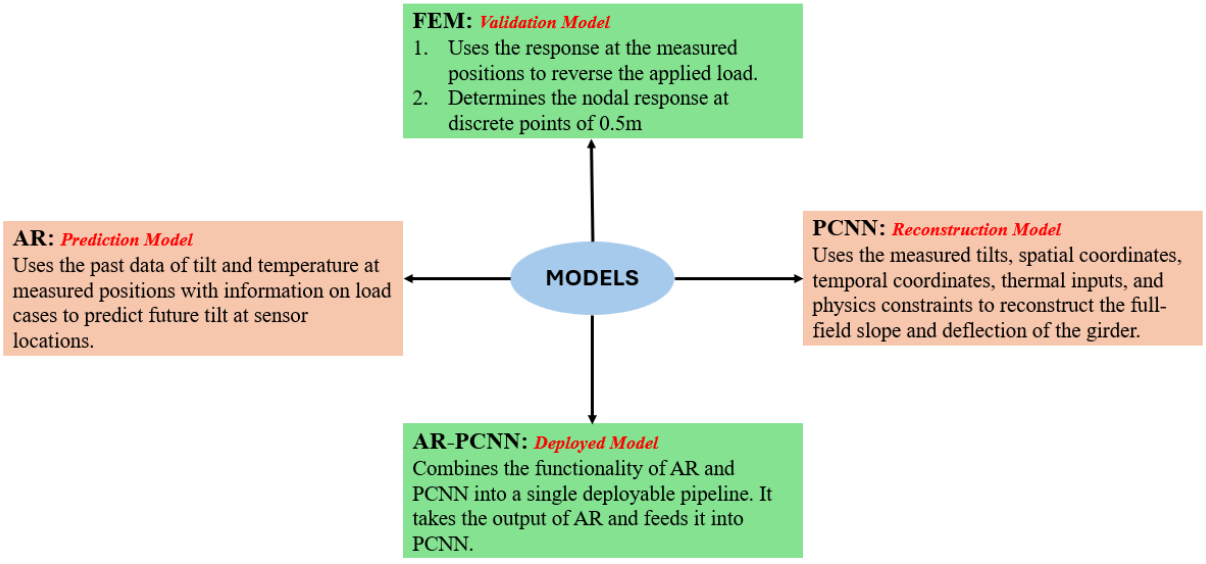


Figure 19: Model Chart

5.1 Autoregressive Model

The Autoregression has an order of 1, i.e., AR (1). The mathematical formulation uses a cross-sensor modelling such that there are two separate models– one per sensor location, and each one carries some information about the other by using the other's past measurements as input. This is the concept of cross-sensor modelling. The motivation for cross-sensor modelling is driven by the fact that both sensors are on the same continuous girder and exposed to the same ambient temperature. So, when one sensor's tilt changes due to thermal loading, the other changes too. The two models are given by:

Sensor in Span AB Model:

$$tilt_{x\_11m}(t) = a_1 \cdot tilt_{x\_11m}(t-1) + a_2 \cdot tilt_{x\_19m}(t-1) + b_1 \cdot T(t) + b_2 \cdot T_{diff}(t) + c_1 \cdot dT_{11}(t) + c_2 \cdot dT_{19}(t) + d_2 \cdot phase_2 + d_3 \cdot phase_3 + d_4 \cdot phase_4 + f \cdot T(t) \cdot phase_4 + d_0 \quad \textbf{(eqn 29)}$$

Sensor in Span BC Model:

$$tilt_{x\_19m}(t) = a_1' \cdot tilt_{x\_11m}(t-1) + a_2' \cdot tilt_{x\_19m}(t-1) + b_1' \cdot T(t) + b_2' \cdot T_{diff}(t) + c_1' \cdot dT_{11}(t) + c_2' \cdot dT_{19}(t) + d_2' \cdot phase_2 + d_3' \cdot phase_3 + d_4' \cdot phase_4 + f' \cdot T(t) \cdot phase_4 + d_0' \quad \textbf{(eqn 30)}$$

Table 4: AR Feature Table

Feature	Description	Category
$\boldsymbol{tilt_{x\_11m}(t-1)}$	measured tilts at x=11m on day t-1	Self-coupling for Span AB, but Cross-coupling for Span BC
$\boldsymbol{tilt_{x\_19m}(t-1)}$	measured tilts at x=19m on day t-1	Self-coupling for Span BC, but Cross-coupling for Span AB
$\boldsymbol{T(t)}$	mean surface temperature on day t	Thermal feature
$\boldsymbol{T_{diff}(t)}$	the day-to-day temperature change i.e. T(t) − T(t−1)	Thermal feature
$\boldsymbol{dT_{11}(t)}$	mean front-back temperature difference at x = 11 on day t	Self-coupling for Span AB, but Cross-coupling for Span BC
$\boldsymbol{dT_{19}(t)}$	mean front-back temperature difference at x = 19 on day t	Self-coupling for Span BC, but Cross-coupling for Span AB

$\boldsymbol{phase_2}$, $\boldsymbol{phase_3}$, and $\boldsymbol{phase_4}$	one-hot encoded loading phases	Loading phase features
$\boldsymbol{T(t) \cdot phase_4}$	Multiple of T(t) and $phase_4$	Thermal and Loading phase features

The coefficients *a* to *f*, and the intercept terms d_0 are learned during model training. Even though both models share the same features, they have independently fitted coefficients (denoted with primes for the 19m model). Each AR (1) model is fitted separately using Ridge regression (L2 regularization). The closed-form regularized least squares, from which the coefficients are obtained, is given by:

$$\min_{\beta}(\|y - X\beta\|^2 + \alpha\|\beta\|^2) \quad \textbf{(eqn 31)}$$

Where y is the measured value, X is the feature matrix, β is the coefficient matrix, and α is the regularization parameter. α is a hyperparameter and chosen as 0.1 to provide mild L2 regularisation.

5.1.1 Train/Test Split and Validation

The splitting is as follows:

- *Training*: 213 observations (February 22 – September 21)
- *Testing:* 40 observations (September 22 – October 31)

All test observations are in Phase 4 (vehicle 9 t), so the model must generalise within a single loading phase to unseen temperature patterns. The model is validated using four accuracy metrics: RMSE, R2, MAE, and Relative L^2 error. The validation analysis includes residual analysis and error distribution analysis.

5.2 Physics – Constrained Neural Network

The PCNN aims to learn a function w(x, t) that represents the beam deflection at any position x ∈ [0, 30] and any observation t ∈ {1, ..., 253}, subject to four constraints:

a) *Data constraint:* The slope dw/dx at x = 11 m and x = 19 m must match the measured tiltx values.

$$\begin{bmatrix} \left.\frac{dw}{dx}\right|_{x=11\,\mathrm{m}} \\ \left.\frac{dw}{dx}\right|_{x=19\,\mathrm{m}} \end{bmatrix} = \begin{bmatrix} tilt_{x11} \\ tilt_{x19} \end{bmatrix}$$

b) *Physics constraint:* The beam equation w'''' + q/EI = 0 must be approximately satisfied at interior collocation points.

$$\frac{d^4w}{dx^4}\left(x_c^{(i)}\right) + \frac{q}{EI}\left(x_c^{(i)}\right) \approx 0, \qquad i = 1,2,\dots,N_c$$

c) *Boundary constraints:* The deflection and slope must satisfy the support conditions.

$$w(0) = \frac{dw}{dx}(0) = 0, \qquad w(15) = \frac{dw}{dx}(15) = 0, \qquad w(30) = 0, \qquad \frac{d^2w}{dx^2}(30) \approx 0$$

d) *Continuity Constraint:* The bending moment and shear force must be continuous across the interior support.

$$\left.\frac{d^2w}{dx^2}\right|_{x=14.7} = \left.\frac{d^2w}{dx^2}\right|_{x=15.3}, \qquad \left.\frac{d^3w}{dx^3}\right|_{x=14.7} = \left.\frac{d^3w}{dx^3}\right|_{x=15.3}$$

The PCNN uses a neural network to parameterise w(x, t) and trains it by minimising a composite loss function that balances data fidelity against physical consistency. To improve numerical stability, the governing equation is nondimensionalised as described in (eqn 15). This balances the residual terms and improves training convergence. The load parameters $q_{AB}(t)$ and $q_{BC}(t)$ are treated as trainable variables rather than learned with a separate NN. They are jointly optimized with the main neural network parameters. Three PCNN architectures are compared:

1. The Single Polynomial Trial Function (SPTF) Architecture
2. The Piecewise Polynomial Trial Function (PPTF) Architecture, and
3. The Pretrained EnvelopeNet Trial Function (PETF) Architecture.

The distinction in the three architectures is in how the trial function is implemented. Everything else, such as the size of the neural network, input features, activation functions, etc., remains the same. Each architecture addresses a specific limitation discovered in the previous one.

Table 5: PCNNs Feature

Feature	Description	Models	Count
x	Position on the bridge ranging from 0-30m	SPTF, PPTF, and PETF	1
t_fourier	Temporal Fourier features transformed with equation 6 using the five frequencies high-lighted in Table 3.	SPTF, PPTF, and PETF	10
T_fourier	Thermal Fourier features transformed with equation 6 using the five frequencies high-lighted in Table 3.	SPTF, PPTF, and PETF	10
Phase_oh	One hot encoder for the loading phase	SPTF, PPTF, and PETF	4

5.2.1 Single Polynomial Trial Function Architecture

The SPTF model uses a polynomial trial function ϕ(x) that exactly satisfies all boundary conditions, which then multiplies the output of the neural network. The trial solution takes the form:

$$w(x,t) = \phi(x) \cdot NN(x, t\_fourier, T\_fourier, phase\_one\_hot) \quad \textbf{(eqn 32)}$$

where NN is a feedforward neural network with 4 hidden layers of 64 neurons each and tanh activation functions. The Fourier features (t_fourier and T_fourier) are used instead of the ordinary normalized features (t_norm and T_norm) because they save the neural network the stress of learning arbitrary functions of t and T. The Fourier variant hands the network the pre-computed multi-scale features, so it can focus on learning the coefficients. The trial function ϕ(x) is constructed by multiplying factors that produce zeros at each boundary condition:

- w(0) = 0 and w'(0) = 0: require a double root at x = 0 → factor x^2
- w(15) = 0 and w'(15) = 0: require a double root at x = 15 → factor $(x - 15)^2$
- w(30) = 0: require a single root at x = 30 → factor $(x - 30)$

These three factors are coupled together to give a degree-5 polynomial that is expressed in (eqn 20). This equation satisfies all boundary conditions except $\frac{d^2w}{dx^2}(30) = 0$ which is added as a loss component (moment loss) in the loss function. The other five boundary conditions are satisfied by the polynomial regardless of the NN output. The loss components in the loss function are simply the data loss, PDE loss, continuity loss, and moment loss.

$$Loss = \lambda_D \cdot L_{Data} + \lambda_{PDE} \cdot L_{PDE} + \lambda_{Cont} \cdot L_{Continuity} + \lambda_{Mom} \cdot L_{Moment} \quad \textbf{(eqn 33)}$$

Where;

λ_D=20, $\lambda_{PDE} = \min\left(0.1 + \frac{\text{current epoch}}{\text{total number of epochs}}, 1.0\right)$, and

$$\lambda_{Cont} = \lambda_{Mom} = \min\left(1 + \frac{20 \cdot \text{current epoch}}{\text{total number of epochs}}, 20.0\right)$$

Below is the detailed step-by-step procedure of the training

1. Define the trial function ϕ(x) = x²(x−15)²(x−30) / PHI_SCALE
2. Prepare the 25 input features as described in Table 5.
3. Initialise the neural network with four hidden layers of 64 neurons each with tanh activation, and one output neuron with Xavier initialisation of gain 0.3
4. Initialise the load parameters q_AB(t) and q_BC(t) as trainable scalars with 5.0 kN/m.
5. Train for 5000 epochs of Adam optimisation with phase-balanced mini-batches of 32 observations. In each epoch:
    a) Feed the 25 input features through the network to obtain the NN output
    b) Multiply the trial function by the NN output to obtain the deflection w(x,t)
    c) Compute the first spatial derivative w'(x,t) to obtain the slope
    d) Compute the data loss L_data as the MSE between −w'×1000 and the measured tiltx at x = 11 m and x = 19
    e) Compute the second spatial derivative w''(x,t) to obtain the bending moment
    f) Compute the third spatial derivative w'''(x,t) to obtain the shear force

g) Compute the continuity loss L_cont by comparing w'' and w''' at x = 14.7 m and x = 15.3 m (across the interior support)

h) Compute the moment loss L_mom as the MSE of w''(29.9), enforcing zero moment at the pinned support

i) Compute the fourth spatial derivative w''''(x,t) at collocation points in both spans

j) Compute the PDE loss L_pde as the MSE of (w'''' + q/EI), using the current load parameter q and the known flexural rigidity EI

k) Combine the losses, L_total as given in equation 12c

l) Backpropagate L_total to update both the NN weights and the load parameters q_AB(t), q_BC(t) simultaneously

6. Refine with 80 steps of L-BFGS quasi-Newton optimisation using the same loss formulation, with strong Wolfe line search for fine convergence.
7. Predict the full field i.e evaluate w(x,t) and w'(x,t) at all spatial positions (0.5 m intervals, 0–30 m) for all 253 observations.
8. Evaluate statistical and physical accuracy by comparing reconstructed slopes and deflections against the FEM baseline at x = [5, 10, 20, 25 m].

This is pictorially shown in Figure 20 and in a more detailed tabular form in APPENDIX 3

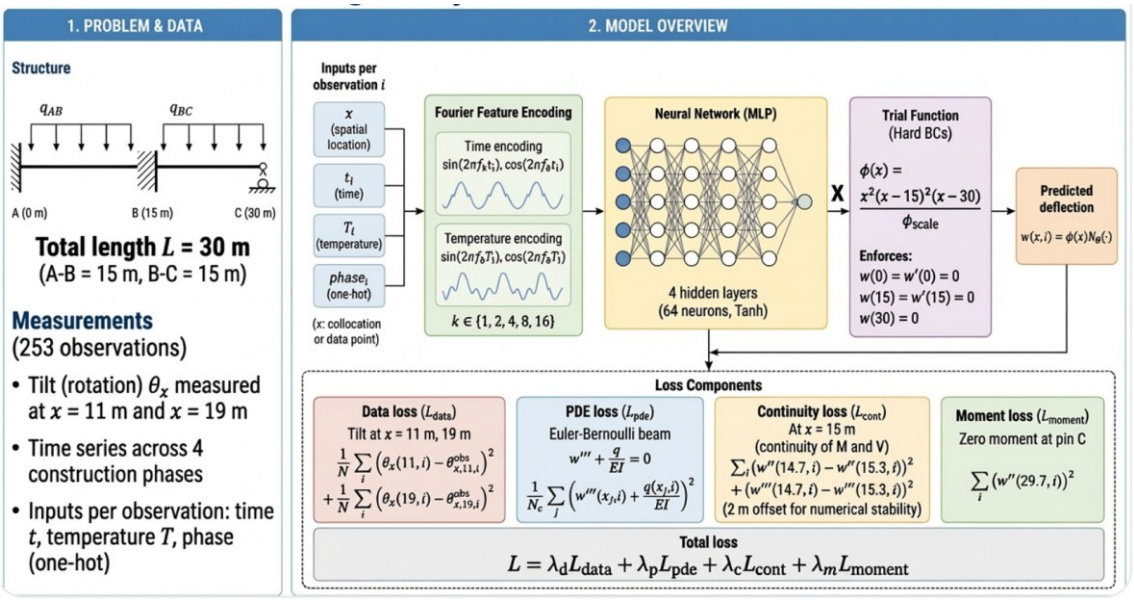


Figure 20: Single Trial Function PCNN

5.2.2 Piecewise Polynomial Trial Function Architecture

SPTF helps enforce hardly constrained boundary conditions through (eqn 20) trial function, but it does not satisfy all boundary conditions simultaneously. Only 5 out of the 6 boundary conditions were enforced directly through the trial function. The remaining condition (the zero moment at the pinned support) was imposed as a soft constraint in the loss function. In addition, using one global trial function tends to bias the solution toward satisfying one span more accurately than the other.

PPTF addresses these limitations by defining a separate trial function for each span with an algorithm that switches the function relative to the span of interest. This allows all boundary conditions to be enforced directly through the trial functions, eliminating the need for the additional moment loss term in the loss function. The training procedure is similar to SPTF, only that the trial function is switched across spans. The detailed workflow of the algorithm is presented in

APPENDIX 4. The trial function of each span is given by:

Span AB (Fixed-Fixed) $\phi_{AB} = x^2(x-15)^2$ **(eqn 34)**

Span BC (Fixed-Pinned) $\phi_{BC} = \xi^2(15-\xi)(22.5-\xi)$ **(eqn 35)**

where $x \in [0,30]$ and $\xi = x - 15$. The loss function is then reduced to:

$$Loss = \lambda_D \cdot L_{Data} + \lambda_{PDE} \cdot L_{PDE} + \lambda_{Cont} \cdot L_{Continuity} \quad \textbf{(eqn 36)}$$

5.2.3 Pretrained EnvelopeNet Trial Function Architecture

PPTF uses two separate polynomials with a hard if/else switch, but their higher derivatives don't match at the mid support. This creates discontinuities that a smooth tanh network cannot represent. PETF addresses this limitation by using a single C∞ function on [0, 30] that transitions smoothly between the two shapes. The smoothness of the function is learnt by a separate pretrained NN, not enforced. This pretrained NN termed “env(x)”, uses low-frequency spatial Fourier features [1, 2, 3] to predict the solution's shape. Env(x) is fed to the main NN as an input to give it distance-to-boundary

information. Similarly, env(x) also multiplies the output of the main NN in a similar way as done with SPTF and PPTF. The final trial solution of PETF architecture is given by:

$$w(x,t) = env(x) \cdot MainNN(x, env(x), t, T, phase) \quad \textbf{(eqn 37)}$$

where env(x) is the pretrained NN that approximates the analytical beam deflection shapes expressed in (eqn 34) and (eqn 35). The algorithm table of PETF is shown in APPENDIX 5

PETF uses a d4 constancy penalty that pushes the envelope's 4th derivative toward a constant within each span during pretraining to overcome spectral bias while avoiding the derivative oscillation caused by high frequencies (amplified by $(2\pi f/L)^4$ in the 4th derivative).

The step-by-step procedure of the pretrained env(x) is as follows:

1. Define the analytical target shapes: ϕ_AB(x) and ϕ_BC(ξ)
2. Initialise EnvelopeNet with a 3-hidden-layer, 64-neuron tanh network with 7 spatial Fourier inputs: [x/L, sin/cos(2πfx/L)] at f = [1, 2, 3].
3. Train for 10,000 epochs of Adam optimisation. In each epoch:
   a) Evaluate E(x) at 2000 uniformly spaced points along [0, 30] m
   b) Compute the shape loss as the MSE between E(x) and the analytical target
   c) Enforce boundary values: E(0) = E(15) = E(30) = 0
   d) Enforce boundary slopes: E'(0) = E'(15) = 0
   e) Enforce zero moment at pin: E''(30) = 0
   f) After epoch 2000, compute E''''(x) at 15 interior points per span and penalise its variance
   g) Combine losses and backpropagate
4. Freeze all EnvelopeNet parameters.

5.3 Autoregressive-PCNN Hybrid

The Autoregressive - PCNN (AR-PCNN) couples the AR predictive model and the PCNN reconstructive model into a single end-to-end pipeline. The pipeline flow as detailed in APPENDIX 6 is as follows:

(Past measured tilt_x + forecast T + known phase) → *AR model* → predicted tilt_x

(predicted tilt_x) → $PCNN$ → {full-field $\text{slope}(x)$& $\text{deflection}(x)$ prediction}

5.4 Finite Element Method

The beam is discretised using Hermite cubic beam elements with 2 DOFs per node (deflection w and slope θ = dw/dx). The element stiffness matrix for length L_e is:

$$K_e = \frac{EI}{L_e^3}\begin{bmatrix} 12 & 6L_e & -12 & 6L_e \\ 6L_e & 4L_e^2 & -6L_e & 2L_e^2 \\ -12 & -6L_e & 12 & -6L_e \\ 6L_e & 2L_e^2 & -6L_e & 4L_e^2 \end{bmatrix}$$

The consistent load vector for downward UDL q is:

$$f_e = \begin{bmatrix} -\frac{qL_e}{2} & -\frac{qL_e^2}{12} & -\frac{qL_e}{2} & \frac{qL_e^2}{12} \end{bmatrix}^T$$

The mesh has 0.5 m spacing with sensor positions both on grid nodes, totalling 61 nodes and 122 DOFs. The global system $[K]\{u\} = \{F\}$ is assembled by the direct stiffness method. The five DOFs are prescribed:

- Support A (x = 0): w = 0, θ = 0 (fixed) → eliminates DOFs 1, 2
- Support B (x = 15): w = 0, θ = 0 (fixed) → eliminates DOFs 61, 62
- Support C (x = 30): w = 0 (pinned), θ free → eliminates DOF 121

This leaves 117 free DOFs. The reduced system $[K_{ff}]\{u_f\} = \{F_f\}$ is solved by direct factorisation. The slope is converted from rad to mm/m via:

$$tilt_x = -\left(\frac{dw}{dx}\right) \times 1000 \quad \text{[mm/m]}$$

Positive tilt corresponds to clockwise rotation, and positive deflection is reported as downward (defl_mm = −w × 1000).

5.4.1 Inverse Analysis for Mean Phase Load Identification

Two load cases are present across all four loading phases: Dead load (gravity load) and prestressed upward tendon force. Dead load is simply the weight of the girder and the weight distributed from the deck.

$$\text{Dead load} = \left(A_{girder} \cdot \gamma_{PC}\right) + \left(\gamma_{RC} \cdot t_{deck} \cdot b_{deck}\right) \cdot \frac{1}{2} = 11.875 kN/m$$

The factor $\frac{1}{2}$ accounts for the middle girder carrying half of the deck load.

5.4.1.1 Loading Phase 1

The only mechanical load present in loading phase 1 is the dead load and tendon load. Since dead load is known, the only unknown load to be discovered is the tendon load. The tendon load can be determined by unit load calibration based on the principle of Finite Linearity. The equation to discover an unknown can be expressed as:

$$q_{discovered} = \frac{\text{mean}\left(measured\_slope(x_s)\right) - slope(x_s)\ from\ q_{known}}{slope_{unit\ response}(x_s)} \quad \textbf{(eqn 38)}$$

Hence, the equivalent distributed tendon load in span AB and BC is:

$$tendon_{AB} = \frac{mean\left(tilt_{x11}^{Ph1}\right) - sg_{11}}{su_{11}} = 9.3454 \text{ kN/m } (78.7\% \text{ of gravity})$$

$$tendon_{BC} = \frac{mean\left(tilt_{x19}^{Ph1}\right) - sg_{19}}{su_{19}} = 4.6530 \text{ kN/m } (39.2\% \text{ of gravity})$$

where $mean\left(tilt_{x11}^{Ph1}\right)$ is the mean measured tilt at

$= 11$ m during loading phase 1 in mm/m,

sg_{11} is the calculated slope due to gravity load in mm/m, and su_{11} is the calculated unit-load slope in mm/m per kN/m.

Consequently, the equivalent load in each phase can be expressed as:

$$q_{equivalent}^{(P)} = q_{known} + q_{discovered} \quad \textbf{(eqn 39)}$$

For instance, the equivalent load in phase 1 is:

$$q_{equivalent}^{(1)} = \text{Dead load} + \text{Tendon load}$$

5.4.1.2 Loading Phase 2,3, and 4

In each phase, there is usually one known load and one unknown load. The known load is the equivalent load of the previous phase, while the unknown load is the extra load added in the phase(to be calculated with (eqn 40)). For instance, the known load in phase 2 is the $q_{equivalent}^{(1)}$ while the unknown is the track load. The algebraic sum of

$q_{equivalent}^{(1)}$ and track load will result in $q_{equivalent}^{(2)}$ which will become the known load for phase 3. This strategy is employed to determine the equivalent load of each loading phase of the girder.

5.4.2 Inverse Identification of Observation-Specific Phase Loads

From (eqn 40), we can compute the expected slope per observation by substituting mean$\big(measured_slope(x_s)\big)$:

$$expected\_slope_i = q_{discovered} \cdot slope_{unit\ esponse}(x_s) + slope_{q_{known}} x_s \quad \textbf{(eqn 40)}$$

The difference in the measured slope and the expected slope is termed slope residual, which can be represented as:

$$residual\_slope_i = expected\_slope_i - measured\_slope_i \quad \textbf{(eqn 41)}$$

Similarly, the residual equivalent observation-specific load, which is as a result of other loads such as thermal, is expressed as:

$$Residual_{UDL} = \frac{Residual_{slope,i}}{Unit_{response}} \quad \textbf{(eqn 42)}$$

Finally, the equivalent observation-specific phase load can be expressed as the algebraic sum of the equivalent phase load and the residual equivalent observation-specific load

$$q_i(t) = q_{equivalent}^{(P)} + Residual_{UDL} \quad \textbf{(eqn 43)}$$

5.4.3 Forward Analysis

The equivalent observation-specific phase load (as in (eqn 43)) is used to perform a forward analysis to estimate the slope and deflection of the girder at the discretized node positions using a superposition method given by:

$$slope(x,t) = -\big[\theta_{dead\ load}(x) + \big(Dead\ load - q_i(t)\big) \cdot \theta_{unit\ response}(x)\big] \times 1000 \quad \textbf{(eqn 44)}$$

$$deflection(x,t) = -\big[w_{dead\ load}(x) + \big(Dead\ load - q_i(t)\big) \cdot w_{unit\ response}(x)\big] \times 1000 \quad \textbf{(eqn 45)}$$

6 Result and Discussions

6.1 FEM

Reconstruction Accuracy

As observed in Figure 21, The reconstruction of the FEM has a machine precision accuracy at the instrumented positions with a max error of 2.78×10^{-16} at 11 m and 4.44×10^{-16} at 19 m. The FEM forward curve (red) lies directly on top of the measured curve (blue) with zero visible deviation. This machine-precision agreement is expected by construction because the calibration and forward formulae are algebraic inverses.

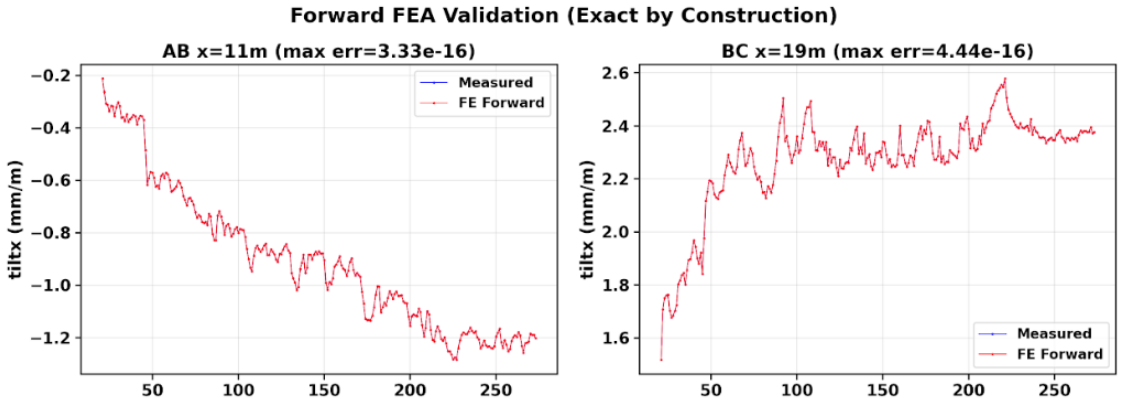


Figure 21: FEM Reconstruction Accuracy

Load Discovery and Tendon Balancing

The stacked bar chart in

Figure **22** reveals the load composition at each phase.

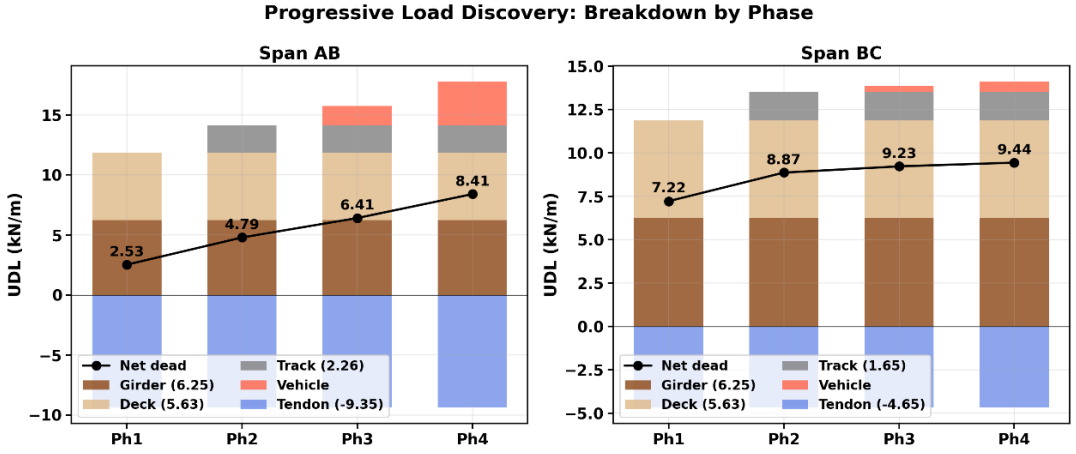


Figure 22: FEM Load Discovery Per-Phase

The tendon provides 9.35 kN/m and 4.65 kN/m uplift in span AB and BC, respectively. This is equivalent to a load balancing of 78.7% in Span AB and 39.2% in Span BC for phase 1. The consequence of this 2:1 ratio is that Span BC will have more deflection and tilt than Span AB, which is evident in the 2D and 3D heatmaps in Figure 28.

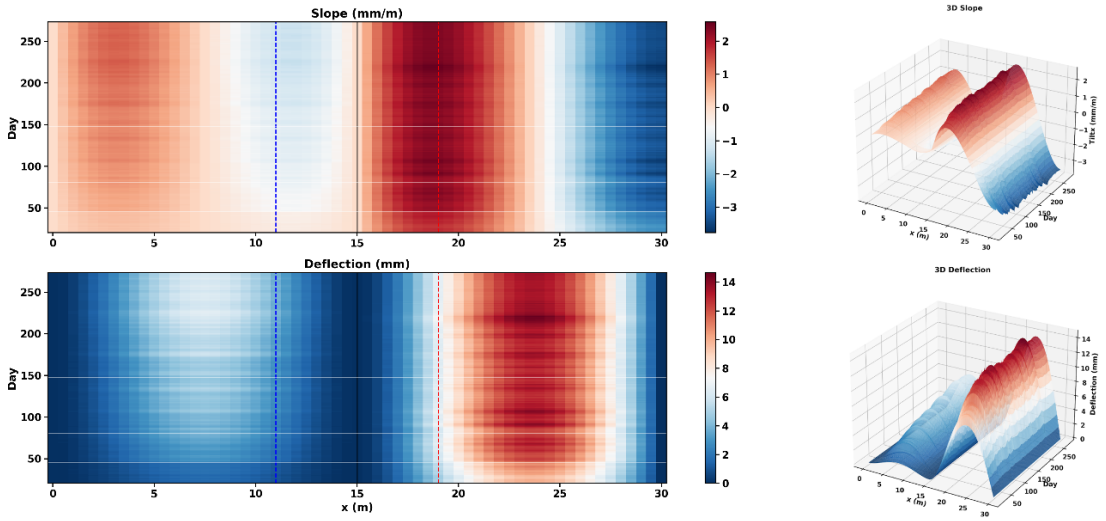


Figure 23: 2D & 3D Heatmap of Full-field Slope and Deflection

As observed in

Figure **22**, the load progression increases across phases, which means the deflection and slope at any day in a specific phase will certainly be greater than that of the previous phase regardless of the season. This is evident in

Figure 24.

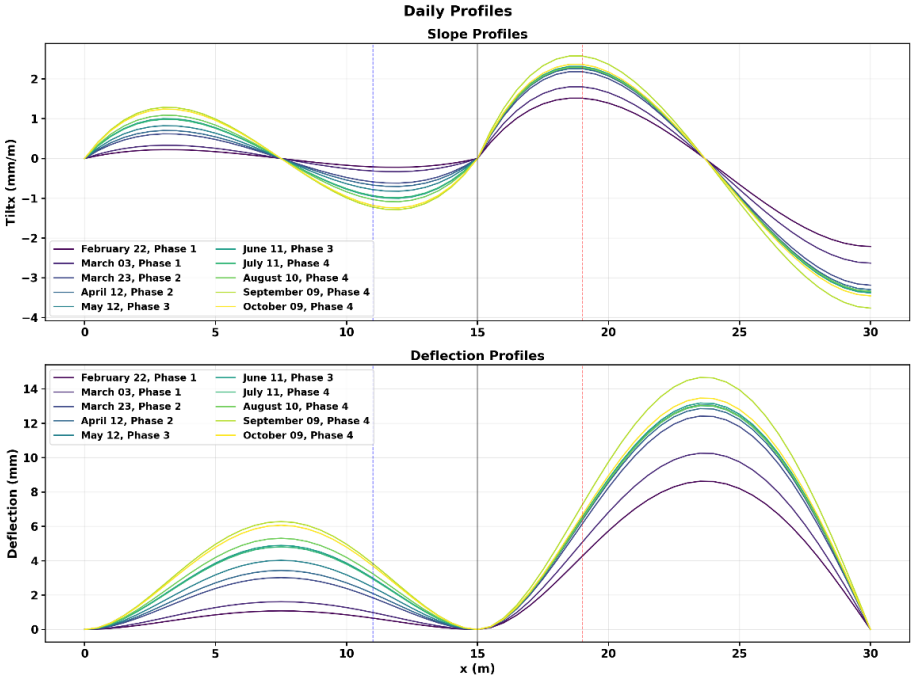


Figure 24: Daily Slope Profile Across all Phases

Seasonal Dependence

As much as the tilt and deflection is dependent on mechanical forces, it is likewise dependent on thermal forcing across seasons. For example, in Phase 4, selected days from summer and autumn show that the magnitude of tilt and deflection is generally higher during autumn under the same loading condition. This occurs because summer warming induces upward thermal bowing that partially counteracts the dead-load deflection, thereby reducing the overall deflection response. In autumn, cooling reduces this thermal uplift, making the mechanical deflection more pronounced.

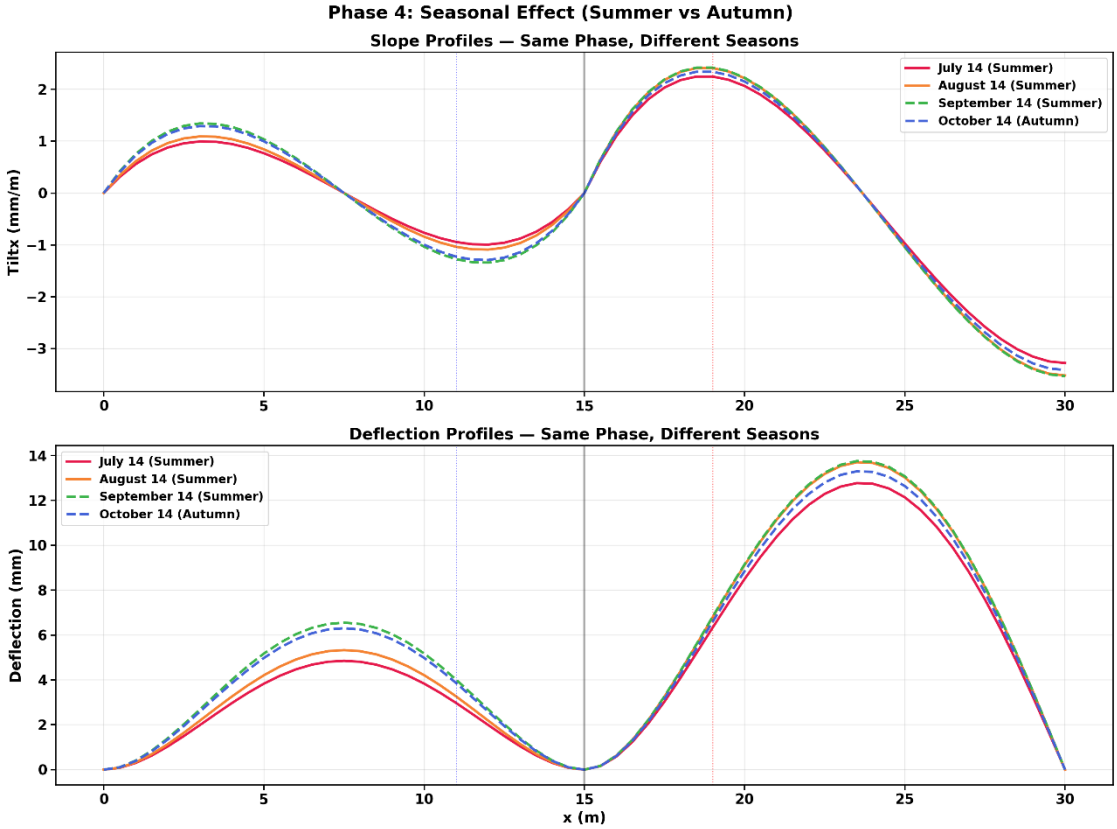


Figure 25: Seasonal Effect on Tilt and Deflection

6.2 PCNN

Model Training Convergence

The convergence plot, Figure 26, shows different behaviours for the three trial functions. In SPTF, all loss components except moment loss reduce to about 10^{-4}, while the moment loss remains around 10^{-2}, indicating difficulty in satisfying the zero-moment condition at the pin support. The PPTF achieves similar data convergence, although the continuity loss remains higher and oscillatory due to the transition at $x = 15$. The PETF gives the most consistent convergence, with all loss terms reaching similar magnitudes between $10^{-3}$and 10^{-4}. In particular, the moment loss reduces significantly compared to SPTF, showing that PETF improves satisfaction of the boundary condition.

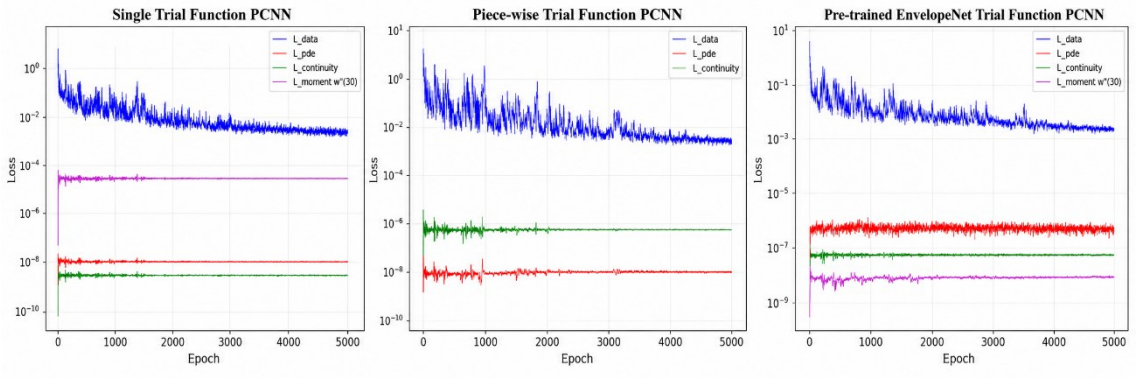


Figure 26: Model Convergence of the PCNNs

Sensor Fit

All three PCNNs achieve a strong sensor fit, confirming the NN's temporal learning is effective regardless of the trial function. Although this is not a guarantee that the model will generalize well at the uninstrumented positions. The generalization of the solution is highly dependent on how the trial function transforms the NN output.

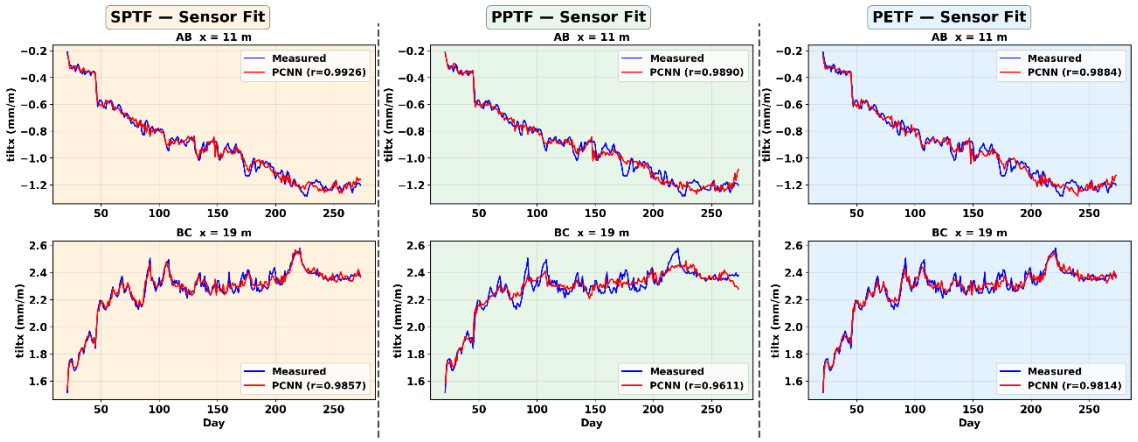


Figure 27: Sensor Fit at Instrumented locations for the PCNN Models

Reconstruction Validation

The generalisation of the three PCNN models is validated against the FEM at four uninstrumented positions x = [5m, 10m, 20m, 25m]. All three models achieve a high Pearson r at most positions, indicating they correctly capture the temporal pattern of the solution. However, each model behaves differently in their ability to capture the correct magnitude.

From

Figure **28**, SPTF generalises well in Span AB (5 m, 10 m) with close tracking of the true solution, but it develops a growing positive bias in Span BC (20 m, 25 m). At 25 m, it even predicts a positive slope where the FEM solution is negative. PPTF performs worse in Span AB due to a symmetric trial function that cannot capture continuous-beam coupling, though Span BC improves slightly. PETF maintains strong correlation across both spans, with a small, uniform offset that does not grow with distance from the sensors and preserves phase transitions and seasonal trends.

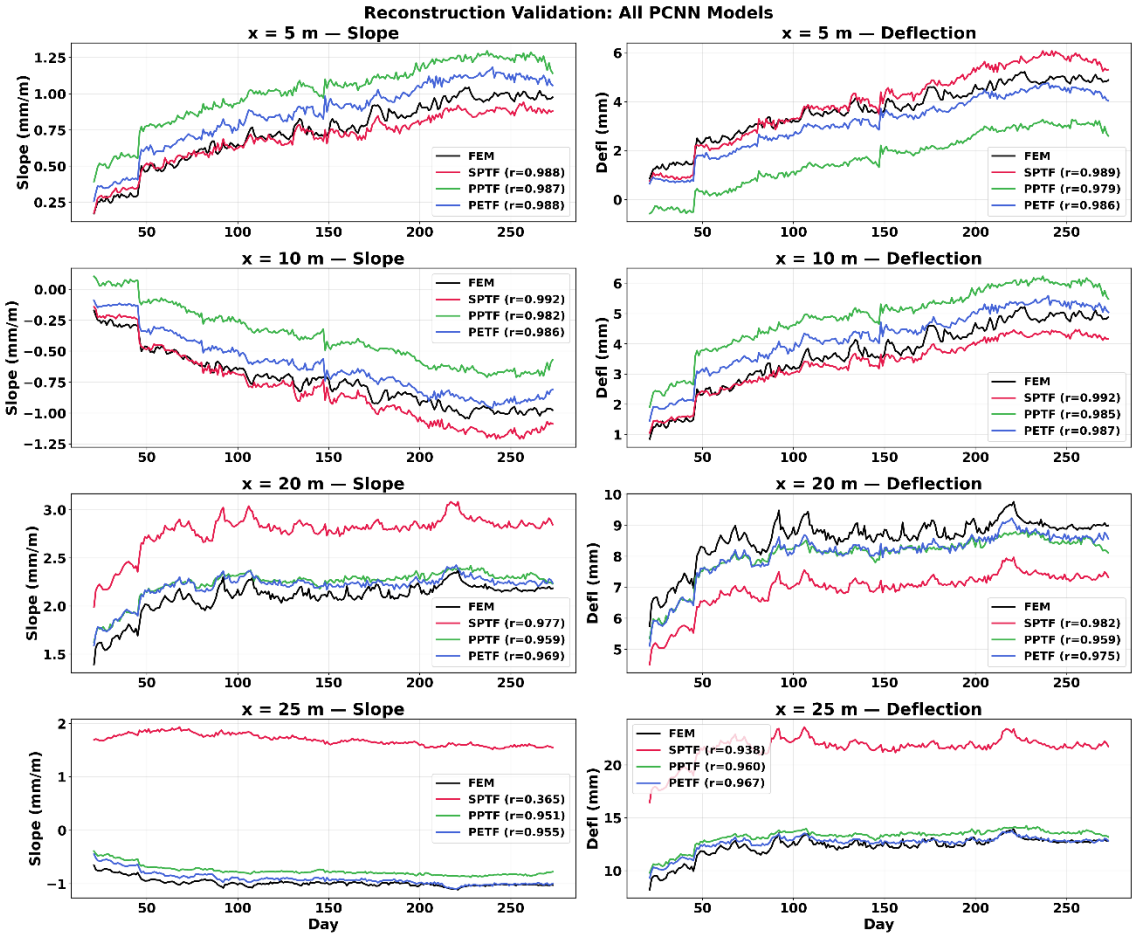
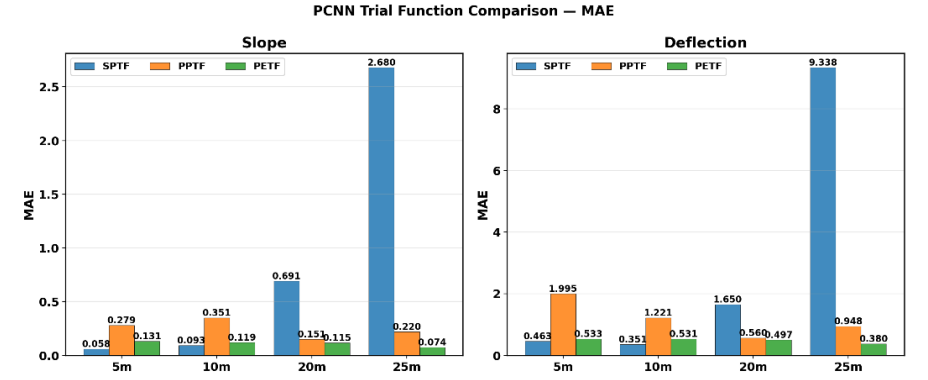
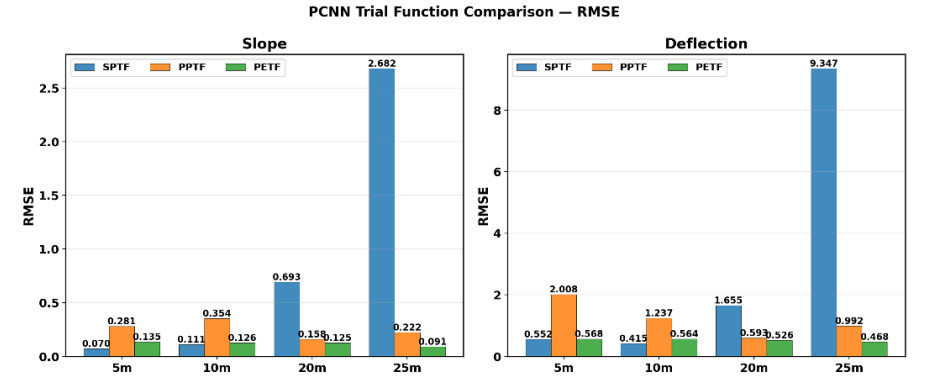
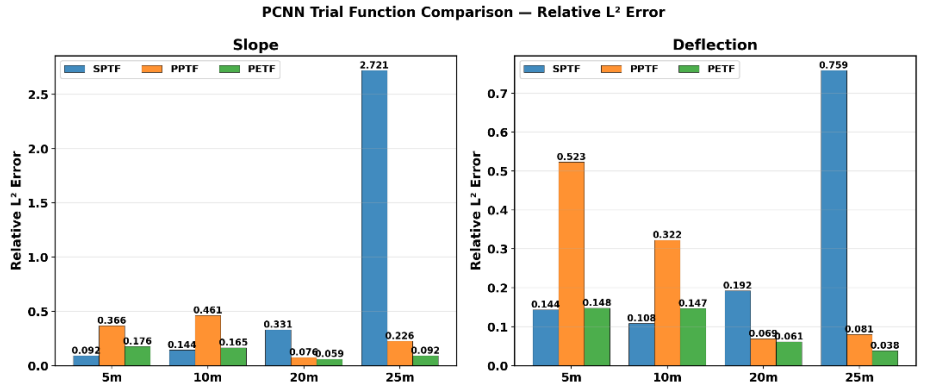


Figure 28: Time series of the PCNNs at Uninstrumented locations.

As observed in Figure 29, SPTF achieves the best span AB performance, but its span BC failure is catastrophic. PETF is the only model with bounded worst-case R^2 (−0.369). At x = 25 m, PETF's slope R^2 is negative, while deflection R^2 is strongly positive. The slope bias exceeds the local slope variance, pushing R^2 negative.


PCNN Trial Function Comparison — MAE
Slope
Deflection
SPTF
PPTF
PETF
MAE
0.058
0.279
0.131
0.093
0.351
0.119
0.691
0.151
0.115
2.680
0.220
0.074
0.463
1.995
0.533
0.351
1.221
0.531
1.650
0.560
0.497
9.338
0.948
0.380
5m
10m
20m
25m


PCNN Trial Function Comparison — RMSE
Slope
Deflection
SPTF
PPTF
PETF
RMSE
0.070
0.281
0.135
0.111
0.354
0.126
0.693
0.158
0.125
2.682
0.222
0.091
0.552
2.008
0.568
0.415
1.237
0.564
1.655
0.593
0.526
9.347
0.992
0.468
5m
10m
20m
25m


PCNN Trial Function Comparison — Relative L² Error
Slope
Deflection
SPTF
PPTF
PETF
Relative L² Error
0.092
0.366
0.176
0.144
0.461
0.165
0.331
0.076
0.059
2.721
0.226
0.092
0.144
0.523
0.148
0.108
0.322
0.147
0.192
0.069
0.061
0.759
0.081
0.038
5m
10m
20m
25m

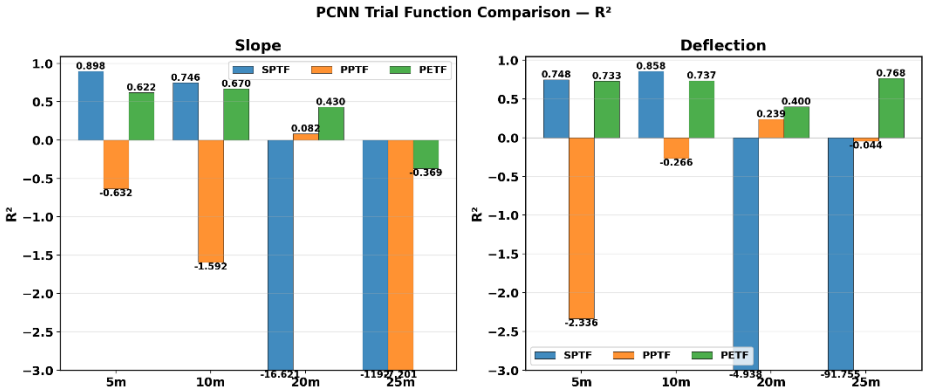

PCNN Trial Function Comparison — R²
Slope
Deflection
SPTF
PPTF
PETF
R²
0.898
-0.632
0.622
0.746
-1.592
0.670
-16.621
0.082
0.430
-1192.201
-0.369
0.748
-2.336
0.733
0.858
-0.266
0.737
-4.938
0.239
0.400
-91.755
-0.044
0.768
5m
10m
20m
25m

Figure 29: Validation of the PCNNs at the Uninstrumented locations

Meanwhile, the same absolute bias is small relative to the deflection magnitude, so deflection R^2 remains high. This demonstrates that a negative R^2 does not necessarily indicate a failed model; it can indicate a good model applied to a low-variance quantity where even small biases dominate. This is better visualized with the mean profile shown in Figure 30.

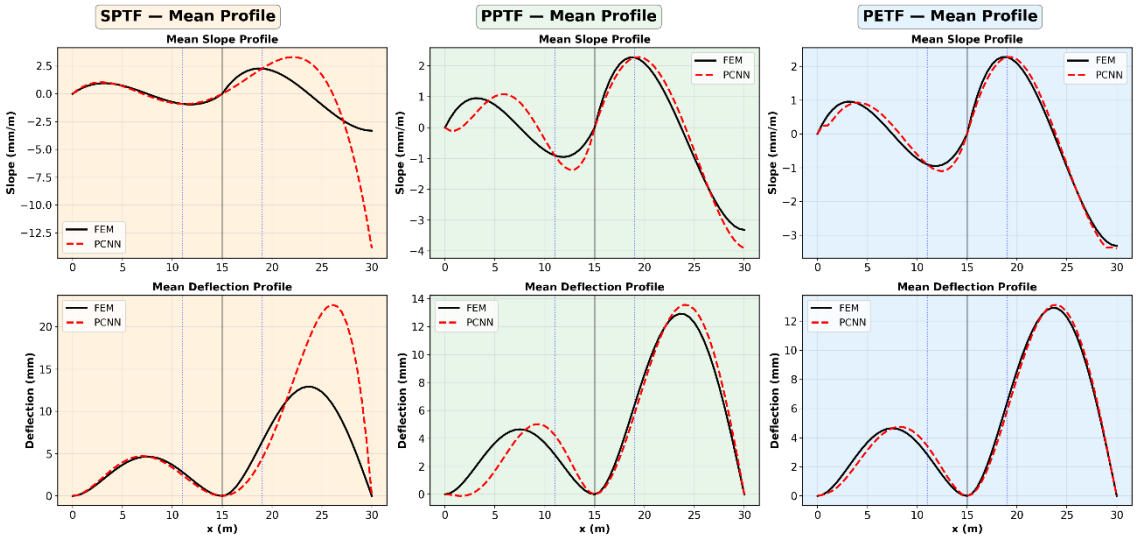


Figure 30: Mean Profile across full span for all PCNNs

Physics Compliance

All three PCNNs have a machine precision boundary condition compliance, especially the polynomial models (SPTF & PPTF), because they enforce the boundary conditions by construction. However, there is a bit of violation for the pinned support moment owing to the fact that it was enforced as a soft loss.

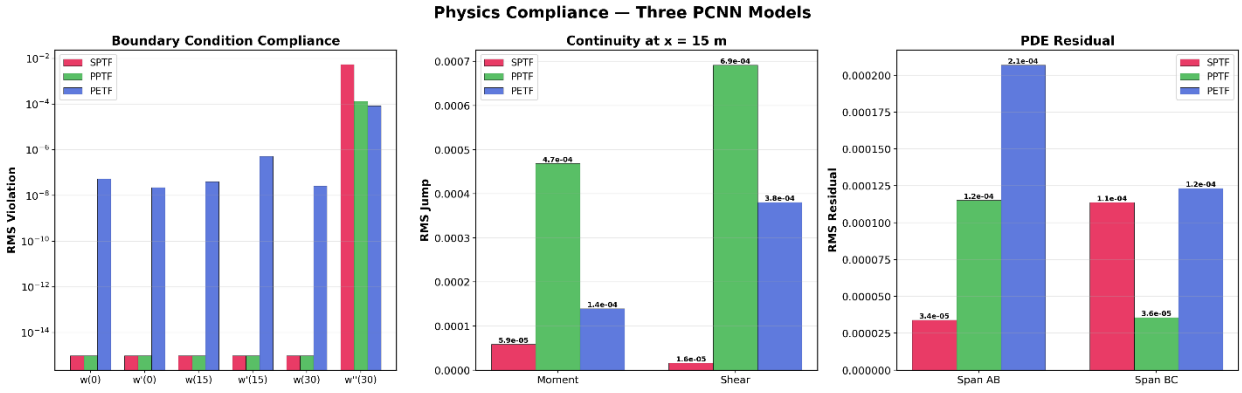


Figure 31: Physics Compliance for all PCNNs

PPTF has the largest continuity violation inherent to the if/else trial function switch. The two polynomials of PPTF have different higher-order derivatives at x = 15, creating jumps that the single smooth NN cannot bridge, regardless of loss weighting. PETF recovers better than PPTF because the C∞ envelope transitions smoothly, but still 2–24× worse than SPTF, whose single smooth polynomial naturally produces continuous derivatives. The PDE residual shows a span-specific pattern. SPTF achieves the best PDE in span AB where its polynomial shape is correct, while PPTF achieves the best in span BC where ϕ_BC is the exact fixed-pinned polynomial. PETF's residuals are marginally higher in both spans because the envelope's fourth derivative is not exactly constant.

Load Discovery

All three models show the same qualitative load discovery behaviour, confirming the limitation is structural (one sensor per span) rather than trial-function-dependent. In span AB, the PCNN's q follows the FEM's phase transitions in the correct direction but with compressed amplitudes, which means the NN captures relative load changes but attenuates absolute levels. In span BC, q is essentially flat at ~5 kN/m regardless of the model or phase, while the FEM ranges 7–10 kN/m. The NN absorbs the phase-to-phase load variation into its spatial modulation, leaving q as a nearly constant offset.

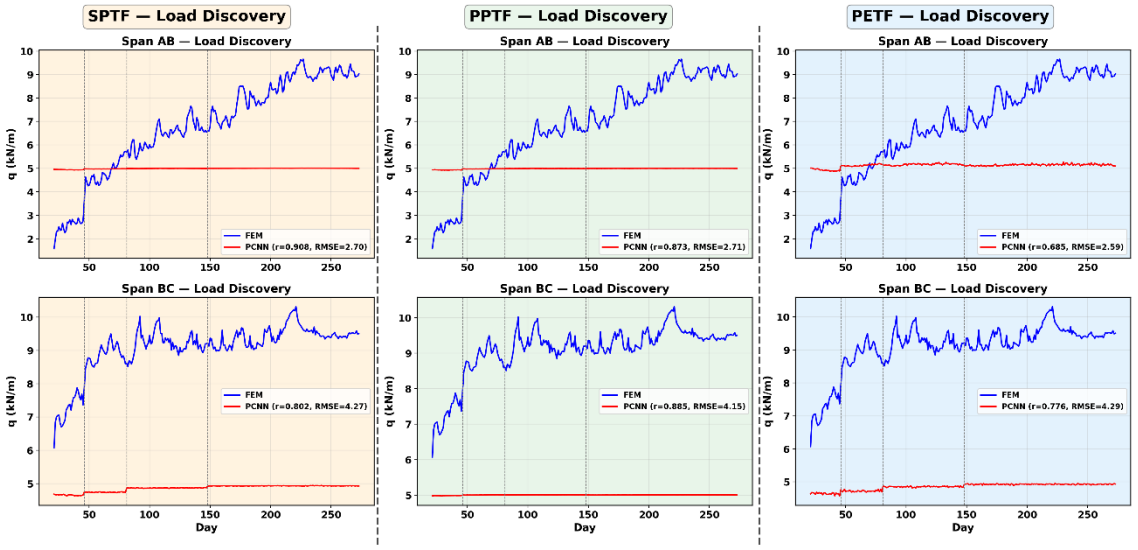


Figure 32: Load Discovery of the PCNNs

Why learn the load q(t) as a parameter instead of a neural network?

An alternative to the per-day scalar parameterisation of q(t) is a dedicated load network q_NN(t, T, phase) → (q_AB, q_BC) that would impose smoothness and reduce the 506 free parameters to a compact architecture. However, a load NN is not ideal because:

1. *Optimization degeneracy*: Two coupled NNs in the PDE loss w'''' + q_NN/EI = 0 can shift blame for the residual, whereby both NNs adjust simultaneously and produce degenerate solutions in which neither the load nor the deflection is physically correct.
2. *Signal structure*: The true net UDL is not a smooth deterministic function of t, T, and phase. A load NN would impose incorrect smoothness.

6.3 AR

Both sensor models achieve a positive R^2 and an RMSE below the mean daily tilt change.

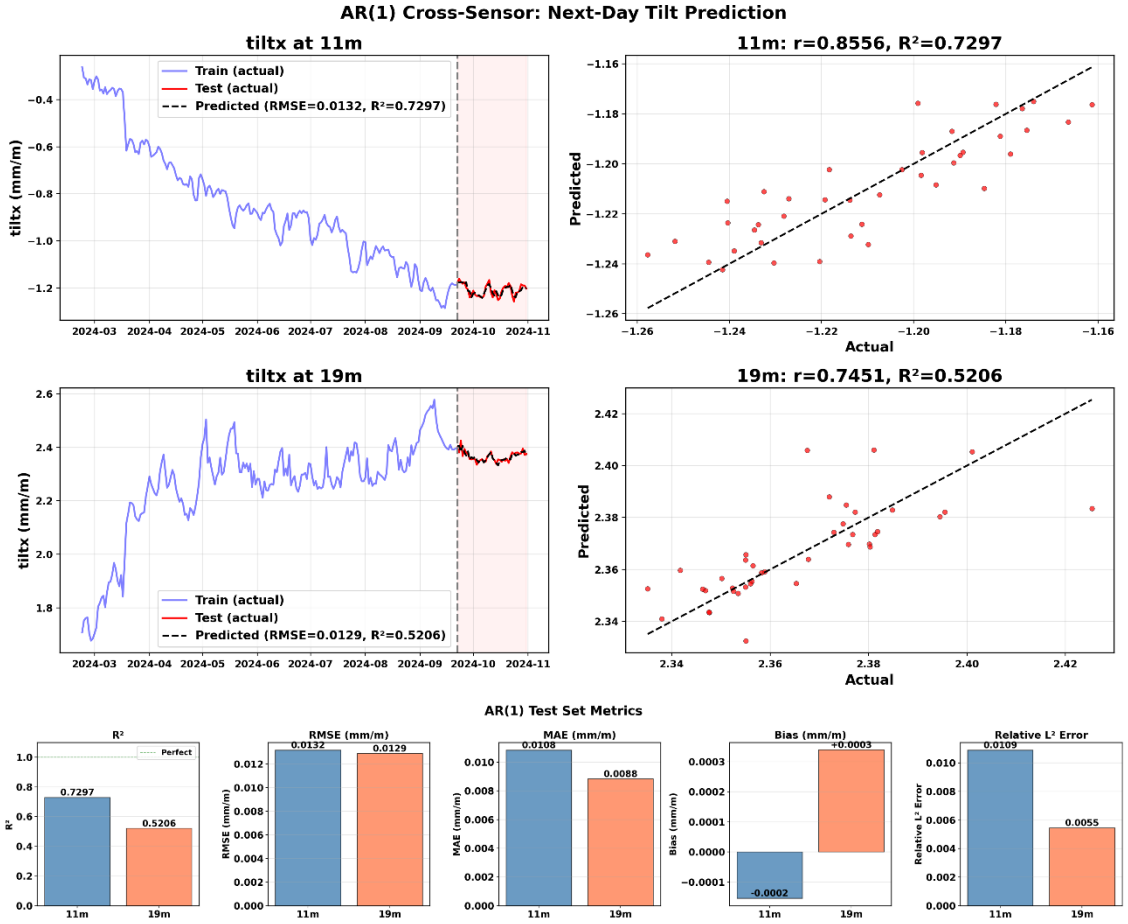


Figure 33: AR Validation Plots

The sensor prediction at 19m has lower RMSE and lower Relative L^2 despite lower R^2. This is because R^2 normalises by variance (which is low in the quiet autumn test period at 19 m), while Relative L^2 normalises by signal magnitude (which is large at 19 m). When the signal-to-noise ratio is evaluated by magnitude rather than variance, the 19 m prediction is actually more accurate.

Coefficient Analysis

Self-coupling dominates both sensors: The *tiltx_11m(t−1)* and *tiltx_19m(t−1)* coefficients dominate the Span AB and Span BC models, respectively. This quantifies that yesterday's tilt explains 66–81% of today's. Aside *T(t),* all other coefficients are sign-flipped between the two models. Phase coefficients are negative at Span AB and positive at Span BC. This is not coincidental — the two sensors sit on opposite sides of the interior support, so every physical effect, such as dead load, thermal curvature, and gradient heating

produces opposite slope signs. The magnitudes differ because the spans have different boundary conditions and stiffness.

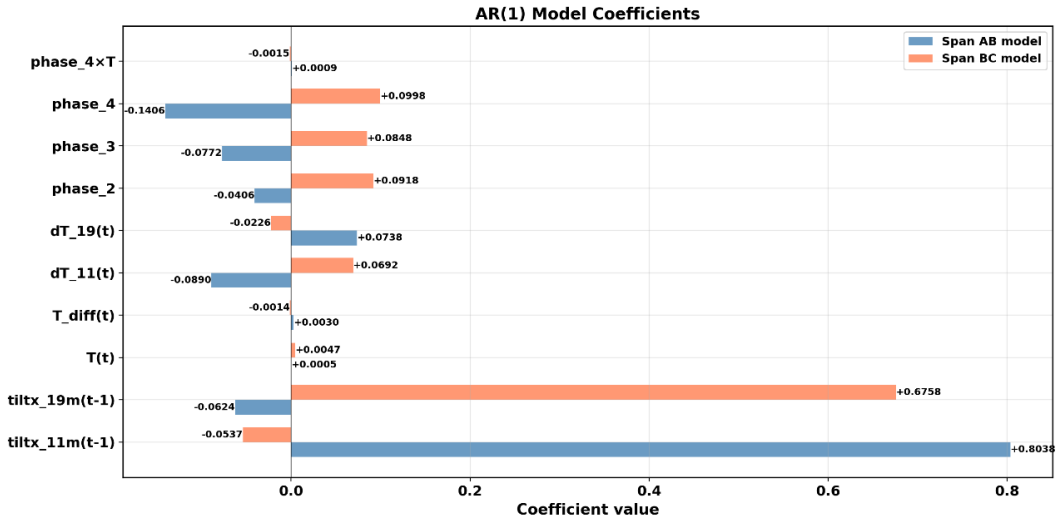


Figure 34: AR Model Coefficients

T(t) and *T_diff(t)* have near-zero coefficients because the autoregressive term already encodes the thermal state. The explicit T coefficient only captures the incremental effect beyond what persistence already provides. Similarly, *phase_4×T* has a near-zero coefficient but is critical for performance. Removing it drops the R^2 of the Span BC model from +0.521 to −0.689. The paradox resolves because the coefficient doesn't need to be large.

Error Structure

The residual-temperature at 11 m shows a correlation of r = −0414. The model overpredicts at high temperatures and underpredicts at low temperatures, indicating remaining nonlinear thermal effects the linear model cannot capture. At 19 m, the correlation is weaker with one outlier near T = 18°C, which is likely a day with anomalous vehicle positioning.

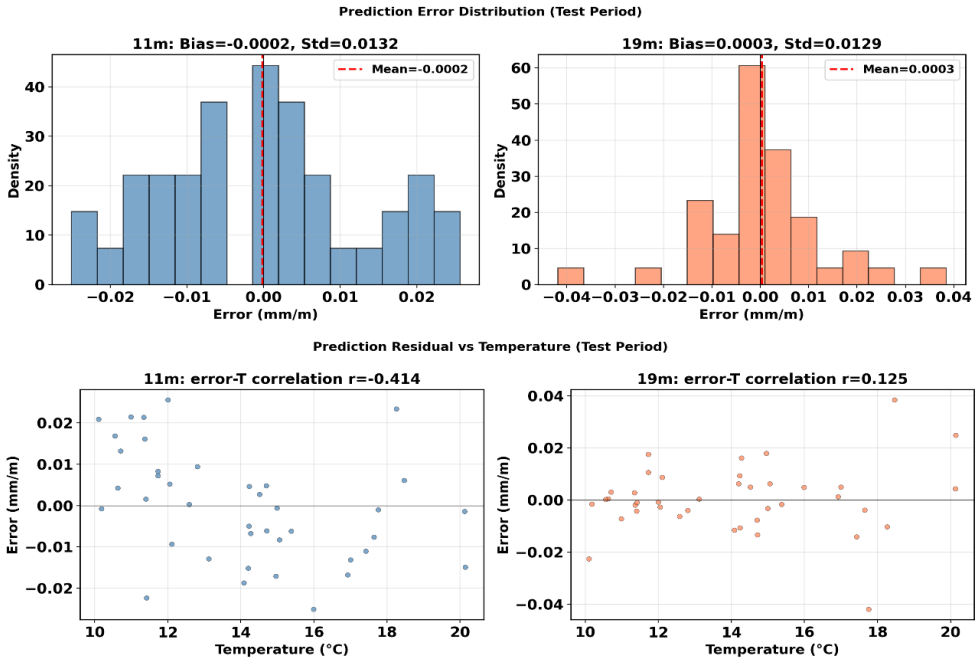


Figure 35: AR Residual Analysis

6.4 AR-PCNN

Comparing the AR-PCNN against PETF reveals a trade-off. Span AB benefits substantially: x = 5 m slope R^2 rises from 0.622 to 0.912, RMSE halves from 0.135 to 0.065 mm/m. At x = 25 m, slope R^2 flips from −0.369 to +0.070 (positive). But span BC partially degrades: at x = 20 m, slope R^2 drops from 0.430 to 0.157, and at x = 25 m, deflection R^2 falls from 0.768 to 0.436.

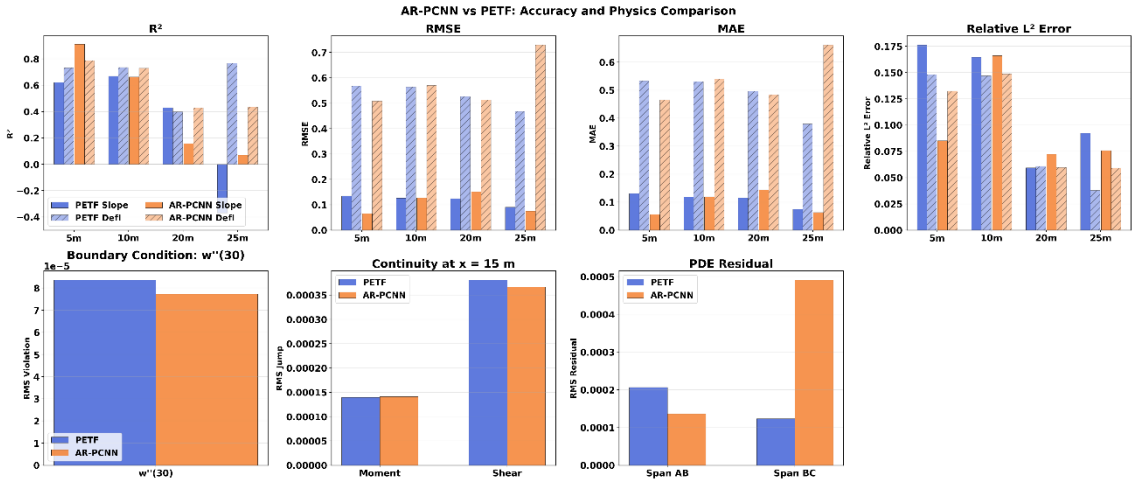


Figure 36: AR-PCNN vs PETF Performance Analysis

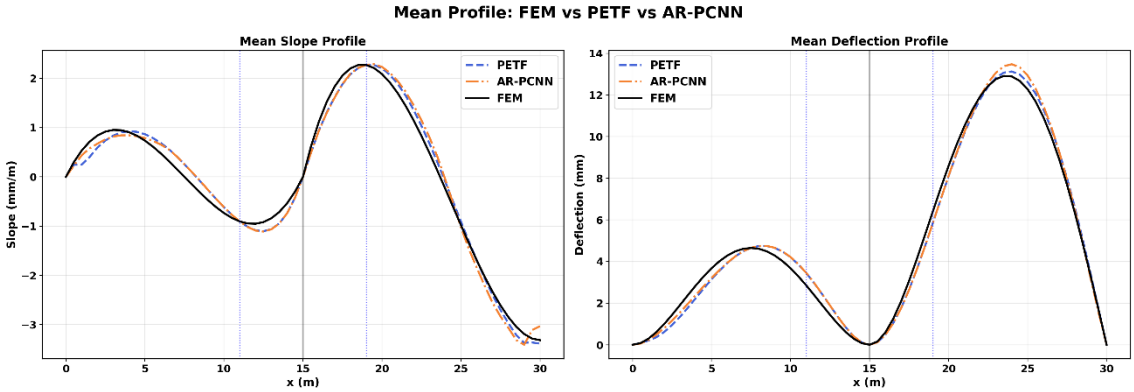


Figure 37: AR-PCNN vs PETF Result

The mechanism for this is attributed to the AR's smoothing effect. The AR predictions have lower day-to-day variance than the measured data. Training on 213 high-variance days + 40 smoothed days biases the optimiser toward a spatial solution with less extreme modulation. This shifts the PETF’s spatial bias in a way that favours span AB at span BC's expense.

7 Conclusions and Recommendations

7.1 Conclusions

This thesis developed a hybrid AR–PCNN framework for next-day tilt prediction at instrumented locations and full-field reconstruction of tilt and deflection at uninstrumented positions on the IDA-KI OpenLab Research Bridge. Three trial functions were evaluated for the PCNN reconstruction task: a single polynomial trial function (SPTF), a piecewise polynomial trial function (PPTF), and a pretrained EnvelopeNet trial function (PETF).

Among them, PETF was the only approach that consistently achieved stable reconstruction across both spans, overcoming the span-dependent failures observed in SPTF and PPTF. For forecasting, a simple AR(1) model with 10 features and a 14-day bias correction outperformed more complex deep learning models, achieving R^2 values of 0.730 and 0.521 at the two sensor locations. When integrated with the PCNN, the resulting AR–PCNN framework enabled next-day full-field prediction of bridge responses, achieving deflection reconstruction accuracy of up to $R^2 = 0.912$.

7.1.1 Specific Conclusions

The following are the specific conclusions of the thesis.

A. Fourier Cyclic Encoding

1. The Fourier temporal encoding was essential for the PCNN's ability to capture smooth seasonal and phase-dependent variation without overfitting. Without it, the NN would need to learn periodicity from raw inputs, requiring a deeper network and more data. More information is presented in APPENDIX 9.

B. Trial Function

1. The PCNNs use the same neural network size and tanh activation function but differ in trial functions. Hence, the accuracy of PCNN as an inverse solver for full-field reconstruction from sparse sensors is governed by the trial function shape, not the NN's temporal capacity.

2. High sensor fit does not guarantee spatial quality. The three PCNNs fit the sensors well but differ by orders of magnitude at non-sensor positions.
3. A single polynomial satisfying all displacement/slope BCs cannot simultaneously encode the correct shape for both a fixed-fixed and a fixed-pinned span.
4. The polynomial trial functions, especially SPTF, would likely perform better if the span BC sensor were positioned at the optimal slope or deflection position. This would've avoided the implementation of PETF, which is more computationally expensive because of the additional pretrained trial function neural network.

C. Physics Constraint

1. The physics constraints (BCs, PDE, and continuity) are necessary but not sufficient. They restrict the solution to plausible beam responses but cannot select the correct one. To do that, accurate sensor data is required.
2. The continuity condition is as important as the boundary condition. PPTF fixes the per-span shape but introduces derivative discontinuities at the mid support.

D. AR & AR-PCNN

1. The *Ridge AR(1)* outperforms *LSTM* (hidden_size = 32, ~13,000 parameters) and *GRU* (hidden_size = 16, ~3,500 parameters) at x = 11 m because the deep models have 100–1000× more parameters than the training samples. Thereby, making overfitting inevitable regardless of regularisation.
2. The AR's smoothing effect (lower day-to-day variance in predictions) acts as implicit regularisation in AR-PCNN. It shifts the solution on the physically compliant manifold rather than finding a universally better point on it.

7.2 Recommendations

1. Repositioning the span BC tiltmeter to the optimal position or adding a second sensor would address the primary cause of span asymmetry and improve reconstruction more than any algorithmic change.
2. For multi-span beams with mixed BCs, pretrained neural network trial functions are recommended over polynomials. Use spatial Fourier frequencies $f \leq 3$ to avoid 4th-derivative amplification.
3. For balanced full-field monitoring, the standalone PETF is preferable to the AR-PCNN due to its more uniform accuracy across spans. The AR-PCNN should be deployed only when next-day prediction is specifically required.

References

Al-Adly, A. I. F., & Kripakaran, P. (2025). Developing physics-informed neural networks for virtual sensing in beams with moving loads. *Engineering Structures*, *338*. https://doi.org/10.1016/j.engstruct.2025.120535

Amaral, P. G. C., & Mazzilli, C. E. N. (2017). Characterization of track geometric imperfections leading to maximal dynamic amplification of internal forces in railway bridges. *Revista IBRACON de Estruturas e Materiais*, *10*(4), 937–956. https://doi.org/10.1590/s1983-41952017000400010

Baez, A., Zhang, W., Ma, Z., Das, S., Nguyen, L. M., & Daniel, L. (2024). *Guaranteeing Conservation Laws with Projection in Physics-Informed Neural Networks*. http://arxiv.org/abs/2410.17445

Bauduin, V., Cuomo, S., & Di Cola, V. S. (2025). Impact of collocation point sampling techniques on PINN performance in groundwater flow predictions. *Journal of Computational Mathematics and Data Science*, *14*. https://doi.org/10.1016/j.jcmds.2024.100107

Charles, J. A., & Skinner, H. D. (2004). Settlement and tilt of low-rise buildings. *Proceedings of the Institution of Civil Engineers - Geotechnical Engineering*, *157*(2), 65–75. https://doi.org/10.1680/geng.2004.157.2.65

Chung, S. W., Castonguay, S., Roy, S., Penwarden, M., Fu, Y., & Roy, P. (2026). *Hard-constrained Physics-informed Neural Networks for Interface Problems*. http://arxiv.org/abs/2604.08453

Cook, W., Barr, P. J., & Halling, M. W. (2015). Bridge Failure Rate. *Journal of Performance of Constructed Facilities*, *29*(3), 4014080. https://doi.org/10.1061/(ASCE)CF.1943-5509.0000571

Danielle Collins. (n.d.). *Motion basics: How to define roll, pitch, and yaw for linear systems*. Retrieved March 28, 2026, from https://www.linearmotiontips.com/motion-basics-how-to-define-roll-pitch-and-yaw-for-linear-systems/

Đạt, T., Matsumoto, Y., & Dang, J. (2023). *A Preliminary Study on Physics-Informed Machine Learning-Based Structure Health Monitoring for Beam Structures* (pp. 490–499). https://doi.org/10.1007/978-3-031-39117-0_50

Deng, Z., Huang, M., Wan, N., & Zhang, J. (2023). The Current Development of Structural Health Monitoring for Bridges: A Review. In *Buildings* (Vol. 13, Number 6). MDPI. https://doi.org/10.3390/buildings13061360

Diaz, E. E. M., Moreno, F. N., & Mohammadi, J. (2009). Investigation of common causes of bridge collapse in Colombia. *Practice Periodical on Structural Design and Construction*, *14*(4), 194 – 200. https://doi.org/10.1061/(ASCE)SC.1943-5576.0000006

Farrar, C. R., & Worden, K. (2007). An introduction to structural health monitoring. *Philosophical Transactions. Series A, Mathematical, Physical, and Engineering Sciences*, *365*(1851), 303–315. https://doi.org/10.1098/RSTA.2006.1928

Faulkner, K., Brownjohn, J. M. W., Wang, Y., & Huseynov, F. (2020). Tracking bridge tilt behaviour using sensor fusion techniques. *Journal of Civil Structural Health Monitoring*, *10*(4), 543–555. https://doi.org/10.1007/s13349-020-00400-9

Geneva, N., & Zabaras, N. (2019). *Modeling the Dynamics of PDE Systems with Physics-Constrained Deep Auto-Regressive Networks*. https://doi.org/10.1016/j.jcp.2019.109056

Ha, D. W., Park, H. S., Choi, S. W., & Kim, Y. (2013). A wireless MEMS-based inclinometer sensor node for structural health monitoring. *Sensors (Switzerland)*, *13*(12), 16090–16104. https://doi.org/10.3390/s131216090

Herbers, M., Bartels, J. H., Richter, B., Collin, F., Ulbrich, L., Al-Zuriqat, T., Chillón Geck, C., Naraniecki, H., Hahn, O., Jesse, F., Smarsly, K., & Marx, S. (2024). openLAB – Eine Forschungsbrücke zur Entwicklung eines digitalen Brückenzwillings. *Beton- Und Stahlbetonbau*, *119*(3), 169–180. https://doi.org/10.1002/best.202300094

Hou, X., Yang, X., & Huang, Q. (2005). Using Inclinometers to Measure Bridge Deflection. *JOURNAL OF BRIDGE ENGINEERING*. https://doi.org/10.1061/ASCE1084-0702200510:5564

Jansen, A., Herbers, M., Richter, B., Walker, M., Jesse, F., & Marx, S. (2025). Monitoring data of the openLAB research bridge – Part 1: Reference condition. *Data in Brief*, *60*. https://doi.org/10.1016/j.dib.2025.111624

Kapoor, T., Wang, H., Nunez, A., & Dollevoet, R. (2024). Physics-Informed Neural Networks for Solving Forward and Inverse Problems in Complex Beam Systems. *IEEE Transactions on Neural Networks and Learning Systems*, *35*(5), 5981–5995. https://doi.org/10.1109/TNNLS.2023.3310585

Kraus, M., & Tatsis, K. (2024, May). *SDF-PINNs: Joining Physics-Informed Neural Networks with Neural Implicit Geometry Representation*.

Lagaris, I. E., Likas, A., & Fotiadis, D. I. (1998). Artificial Neural Networks for Solving Ordinary and Partial Differential Equations. In *IEEE TRANSACTIONS ON NEURAL NETWORKS* (Vol. 9, Number 5).

Lee, G. C., Mohan, S., Huang, C., & Fard, B. N. (2013). *A study of US bridge failures (1980-2012)*. State University of New York at Buffalo, New York.

Lu, L., Pestourie, R., Yao, W., Wang, Z., Verdugo, F., & Johnson, S. G. (2021). Physics-Informed Neural Networks with Hard Constraints for Inverse Design. *SIAM Journal on Scientific Computing*, *43*(6), B1105–B1132. https://doi.org/10.1137/21M1397908

Mammeri, S., Barros, B., Conde-Carnero, B., & Riveiro, B. (2025). From traditional damage detection methods to Physics-Informed Machine Learning in bridges: A review. In *Engineering Structures* (Vol. 330). Elsevier Ltd. https://doi.org/10.1016/j.engstruct.2025.119862

Manavi, S., Fattahi, E., & Becker, T. (2024). A trial solution for imposing boundary conditions of partial differential equations in physics-informed neural networks. *Engineering Applications of Artificial Intelligence*, *127*. https://doi.org/10.1016/j.engappai.2023.107236

Millar, B., Hamill, G., Taylor, S., & Robinson, D. (2024). ArchIMEDES: Computer Vision Tracking of the Inherent Changes to Structural Stability of Masonry Arch Bridges Resulting from Increased Bed Scour. *11th European Workshop on Structural Health Monitoring, EWSHM 2024*. https://doi.org/10.58286/29586

Mozumder, R. S., Debnath, R., Rahman, F., Ahmed, A. A., & Alam, M. J. Bin. (2026). Field Validation of a Low-Cost IoT-Based System for Real-Time Monitoring of Hydrologic and Physical Behavior of Slope. *2026 International Conference on Computing, Networking and Communications (ICNC)*, 309–315. https://doi.org/10.1109/ICNC68183.2026.11416850

Ni, P., Li, Y., Sun, L., & Wang, A. (2022). Traffic-induced bridge displacement reconstruction using a physics-informed convolutional neural network. *Computers and Structures*, *271*. https://doi.org/10.1016/j.compstruc.2022.106863

openLAB — Institute of Concrete Structures — TU Dresden. (2023). https://tu-dresden.de/bu/bauingenieurwesen/imb/forschung/grossprojekte/openLAB?set_language=en

Raissi, M., Perdikaris, P., & Karniadakis, G. E. (2018). *Physics-Informed Neural Networks: A Deep Learning Framework for Solving Forward and Inverse Problems Involving Nonlinear Partial Differential Equations*. https://www.sciencedirect.com/science/article/pii/S0021999118307125

Rezaei, S., Harandi, A., Moeineddin, A., Xu, B.-X., & Reese, S. (2022). *A mixed formulation for physics-informed neural networks as a potential solver for engineering problems in heterogeneous domains: comparison with finite element method*. https://doi.org/10.1016/j.cma.2022.115616

Richter, B., Messerer, D., Herbers, M., Speck, K., Laukner, J., Gläser, C., Jesse, F., & Marx, S. (2024). Monitoring of a prestressed bridge girder with integrated distributed fiber optic sensors. *Procedia Structural Integrity*, *64*, 1208–1215. https://doi.org/10.1016/j.prostr.2024.09.168

Sanli, A. K., Uzgider, E. A., Caglayan, O. B., Ozakgul, K., Bien, J., Sanli, A. K., Uzgider, E. A., Caglayan, O. B., & Ozakgul, K. (2000). *Testing Bridges by Using Tiltmeter Measurements*.

Shenton, H., Fernandez, M., Ramanna, N., Chajes, M., Wenczel, G., & Al-Khateeb, H. (2015). *Structural Health Monitoring of a Cable-Stayed Bridge: Using Tiltmeter Data to Determine Edge Girder Deflections*.

SMITH, D. W. (1976). BRIDGE FAILURES. *Proceedings of the Institution of Civil Engineers*, *60*(3), 367–382. https://doi.org/10.1680/iicep.1976.3389

Sonbul, O. S., & Rashid, M. (2023). Algorithms and Techniques for the Structural Health Monitoring of Bridges: Systematic Literature Review. In *Sensors* (Vol. 23, Number 9). MDPI. https://doi.org/10.3390/s23094230

Srl, S. (n.d.). *D-Tilt meters - Digital MEMS tilt meters*. Retrieved www.sisgeo.com/faq

Tang, Y., Cang, J., Zheng, B., & Tang, W. (2024). Deflection Monitoring Method for Simply Supported Girder Bridges Using Strain Response under Traffic Loads. *Buildings*, *14*(1). https://doi.org/10.3390/buildings14010070

Wang, S., Teng, Y., & Perdikaris, P. (2020). *Understanding and mitigating gradient pathologies in physics-informed neural networks*. http://arxiv.org/abs/2001.04536

Wardhana, K., & Hadipriono, F. C. (2003). Analysis of recent bridge failures in the United States. *Journal of Performance of Constructed Facilities*, *17*(3), 144 – 150. https://doi.org/10.1061/(ASCE)0887-3828(2003)17:3(144)

Appendices

APPENDIX 1 The full code of the framework is available on Ridwantican GitHub

APPENDIX 2 Autoregressive Algorithm Table

AR(1) Cross-Sensor with Ridge Regression

Input: $s_{11}(t)$, $s_{19}(t) \in \mathbb{R}^N$ (daily tilt, N=253 days), T(t), $dT_{11}(t)$, $dT_{19}(t)$, phase(t) ∈ {1,2,3,4}, $\alpha = 0.1$

Output: $\hat{s}_{11}(t)$, $\hat{s}_{19}(t)$ for test days t = [214..253], Coefficients β_{11}, $\beta_{19} \in \mathbb{R}^{10}$, metrics on test set

Feature Engineering

For each target day t = 2..N:

Split: X_train = x(2..214), X_test = x(215..253) and y_train = s(2..214), y_test = s(215..253)

Model Training (per sensor k ∈ {11m, 19m})

$\beta\_k = \text{argmin}_\beta \{||y\_train - X\_train \cdot \beta||^2_2 + \alpha \cdot ||\beta||^2_2\}$

Closed-form solution: $\beta\_k = (X^T X + \alpha I)^{-1} X^T y\_train$

Prediction

For test days t = 215..253:

$\hat{s}\_k(t) = \beta\_k_{,0} + \Sigma_j \beta\_k_{,j} \cdot x_j(t)$

Evaluation

For k ∈ {11m, 19m}:

$R^2 = 1 - \Sigma(y-\hat{y})^2/\Sigma(y-\bar{y})^2$; RMSE = $\sqrt{}$(mean($(y-\hat{y})^2$)); MAE = mean($|y-\hat{y}|$); L2 = $\sqrt{\Sigma(y-\hat{y})^2}/\sqrt{\Sigma y^2}$

APPENDIX 3 SPTF Algorithm

Single Polynomial Trial Function PCNN

Input: $s_{11}(t)$, $s_{19}(t) \in \mathbb{R}^N$ (tilt at sensors, N=253), t, T(t), phase(t) ∈ {1,2,4,8,16}, EI, L=30m.

Output: Neural field u(x,t), θ(x,t) ∀x∈[0,30]m, Discovered loads q_AB(t), q_BC(t) $\in \mathbb{R}^N$

Architecture

Trial function: $\hat{u}(x,t) = \varphi(x) \cdot NN(\xi(x,t))$

$\varphi(x) = x^2(x-15)^2(x-30) / \varphi_{max}$ (hard BCs: w=w'=0 at 0,15,30)

Input $\xi(x,t) \in \mathbb{R}^{25}$:

x, sin(2πf·t_norm), cos(2πf·t_norm) for f ∈{1,2,4,8,16}, [×2 for T], one-hot phase $\in \mathbb{R}^4$

NN: 4 hidden layers × 64 neurons, Tanh activation

Learnable: θ_NN (weights), q_AB(t), q_BC(t) (per-observation loads)

Loss Function (Minibatch Size B=32)

$L = 20 \cdot L_{data} + \lambda_p \cdot L_{pde} + \lambda_c \cdot (L_{continuity} + L_{moment})$

$L_{data} = MSE(-\partial\hat{u}/\partial x \cdot 1000, s_{measured})$ at x∈{11,19}m

$L_{pde} = MSE(\partial^4\hat{u}/\partial x^4 + q/EI, 0)$ // evaluated at 20 collocation pts per span

$L_{continuity} = MSE(\partial^2\hat{u}/\partial x^2|_{14.7} - \partial^2\hat{u}/\partial x^2|_{15.3}) + MSE(\partial^3\hat{u}/\partial x^3|_{14.7} - \partial^3\hat{u}/\partial x^3|_{15.3})$

$L_{moment} = MSE(\partial^2\hat{u}/\partial x^2|_{29.9}, 0)$ // soft pin BC

Annealing: $\lambda_p = \min(0.1 + t/5000, 1.0)$ and $\lambda_c = \min(1 + 20t/5000, 20.0)$

Training (Adam 5000 epochs + L-BFGS 80 steps)

For epoch t = 1..5000:

Sample minibatch via weighted sampling (inverse phase frequency)

Compute L via automatic differentiation (4th-order gradients)

Clip gradients $\|\nabla\| \leq 5.0$

Update θ_NN (lr=1e-3), q_AB, q_BC (lr=0.3)

Cosine annealing lr → 1e-6

L-BFGS finetuning: 80 steps, strong Wolfe line search, lr=0.3

Prediction

For any (x,t): $\theta(x,t) = -\partial\hat{u}/\partial x \cdot 1000$, $v(x,t) = -\hat{u} \cdot 1000$

Validation (against FEM baseline at held-out positions)

For x ∈ {5, 10, 20, 25}m, for all t:

θ_PCNN(x,t) = $-\partial\hat{u}/\partial x \cdot 1000$, v_PCNN(x,t) = $-\hat{u} \cdot 1000$; θ_FEM(x,t), v_FEM(x,t)

APPENDIX 4 PPTF Algorithm

Figure 38 below shows the pictorial representation of the PPTF Algorithm, while the succeeding table highlights the step-by-step algorithm workflow.

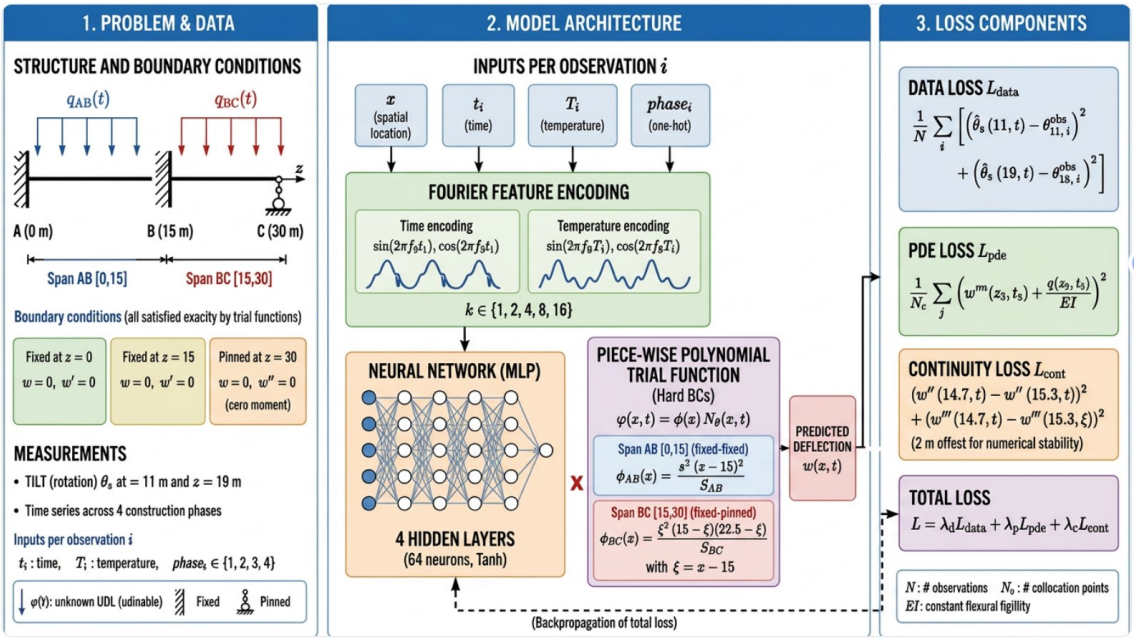


Figure 38: Piece-wise Trial Function PCNN

Piece-wise Polynomial Trial Function PCNN

Input: $s_{11}(t)$, $s_{19}(t) \in \mathbb{R}^N$ (tilt at sensors, N=253), T(t), phase(t), EI = 1.928×10^5 kN·m²

Output: Neural displacement field û(x,t), slope θ(x,t), load q_AB(t), q_BC(t)

Piecewise Trial Functions (all 8 BCs enforced exactly)

Span AB [0,15]: $\varphi_{AB}(x) = x^2(x-15)^2 / S_{AB}$

Span BC [15,30]: $\varphi_{BC}(x) = \xi^2(15-\xi)(22.5-\xi) / S_{BC}$ with ξ=x-15

Displacement: $\hat{u}(x,t) = \varphi_{span}(x) \cdot NN(\xi(x,t))$

Input $\xi \in \mathbb{R}^{25}$: x, sin/cos(2πf·t_norm), sin/cos(2πf·T_norm) for f∈{1,2,4,8,16}, phase_OH∈$\mathbb{R}^4$

NN: 4 hidden layers × 64, Tanh

Loss Function (batch size B=32, collocation points per span C=10)

L = 20·L_data + λ_pde·L_pde + λ_cont·L_continuity

L_data = MSE($-\partial\hat{u}/\partial x\cdot1000$, s_measured) at x=11m (AB), x=19m (BC)

L_pde = MSE($\partial^4\hat{u}/\partial x^4$ + q_span/EI, 0) // C collocation pts per span

L_continuity = MSE($\partial^2\hat{u}/\partial x^2|_{13} - \partial^2\hat{u}/\partial x^2|_{17}$) + MSE($\partial^3\hat{u}/\partial x^3|_{13} - \partial^3\hat{u}/\partial x^3|_{17}$)

Annealing: λ_pde = min(0.1 + t/5000, 1.0) and λ_cont = min(1 + 20t/5000, 20.0)

Training (Adam 5000 + L-BFGS 80)

Learnable parameters: θ_NN (network weights), q_AB(t), q_BC(t) ∈ $\mathbb{R}^N$

Weighted sampling: p(t) ∝ 1/|phase(t)| (inverse phase frequency)

For epoch t = 1..5000:

Sample batch; compute w, $\partial w/\partial x$, ..., $\partial^4 w/\partial x^4$ via autograd

L ← 20·L_data + λ_pde·L_pde + λ_cont·L_continuity

Clip gradients $\|\nabla\theta_{NN}\| \le 5.0$

Update: θ_NN (lr=1e-3), q (lr=0.3), cosine annealing → 1e-6

L-BFGS finetuning: 80 steps, strong Wolfe search

Prediction

for any x ∈ [0,30]:

span ← AB if x≤15 else BC

$\hat{u} = \varphi_{span}(x) \cdot NN(\xi)$; $\theta = -\partial\hat{u}/\partial x\cdot1000$; $v = -\hat{u}\cdot1000$

APPENDIX 5 PETF Algorithm

Figure 39 below shows the pictorial representation of the PPTF Algorithm, while the succeeding table highlights the step-by-step algorithm workflow.

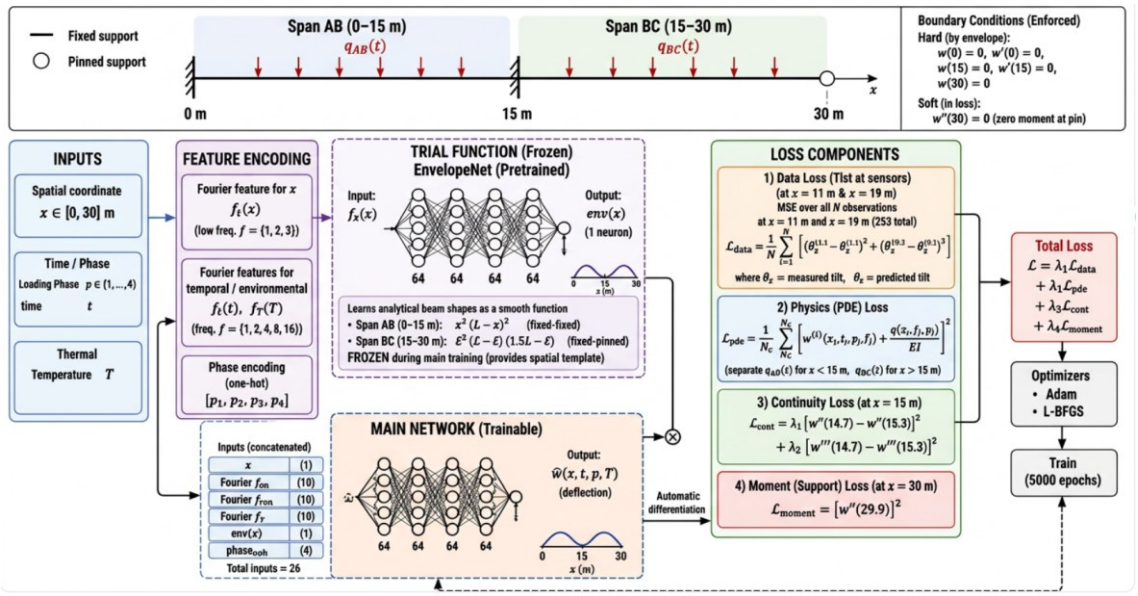


Figure 39: Pre-trained EnvelopeNet Trial Function PCNN

Pre-trained EnvelopeNet Trial Function PCNN

Input: $s_{11}(t)$, $s_{19}(t) \in \mathbb{R}^N$ (tilt at sensors, N=253), T(t), phase(t), EI = 1.928×10^5 kN·m²

Output: Neural field û(x,t), θ(x,t), loads q_AB(t), q_BC(t)

Stage 1: Pretrain Envelopenet (frozen during main training)

env(x): $\mathbb{R} \to \mathbb{R}$, input = [x/L, sin/cos(2πf·x/L)] for f∈{1,2,3}

Architecture: 3 hidden layers × 64, Tanh

Target shape (piecewise analytical):

AB [0,15]: env(x) ≈ x²(15-x)² / S_AB

BC [15,30]: env(x) ≈ ξ²(15-ξ)(22.5-ξ) / S_BC

Loss = MSE(env(x), target) + 500·BC_terms + λ_d4·Var(env'''')

Train 10,000 epochs, Adam (lr=3e-3), freeze all parameters

Stage 2: Main PCNN Training

Trial function: û(x,t) = env(x) · NN(x, env(x), t_features, T_features, phase)

NN: 4 hidden layers × 64, Tanh; input ∈ $\mathbb{R}^{26}$

Learnable: θ_NN, q_AB(t), q_BC(t) ∈ $\mathbb{R}^N$

Loss Function (batch B=32)

L = 20·L_data + λ_pde·L_pde + λ_cont·(L_continuity + L_moment)

λ_pde = min(0.1 + t/5000, 1.0), λ_cont = min(1 + 20t/5000, 20.0)

Training (Adam 5000 + L-BFGS 80)

Weighted sampling ∝ 1/|phase|, gradient clip $\|\nabla\| \le 5.0$

θ_NN: lr=1e-3; q_AB, q_BC: lr=0.3; cosine annealing → 1e-6

Validation (against FEM baseline at held-out positions)

For x ∈ {5, 10, 20, 25}m, for all t:

θ_PCNN(x,t) = -∂û/∂x·1000, v_PCNN(x,t) = -û·1000

θ_FEM(x,t), v_FEM(x,t)

Compute per-position and full-field metrics: r, R², RMSE, MAE, Relative L^2, Bias vs FEM

Load validation: q_PCNN(t) vs q_FEM(t) per span: r, RMSE, Relative L^2

Physics Verification

BC compliance: RMS of w(0), w'(0), w(15), w'(15), w(30), w''(30)

Continuity at x=14.7/15.3; PDE residual per span: RMS and max of $\partial^4\hat{u}/\partial x^4 + q/EI$

APPENDIX 6 AR-PCNN Algorithm

Figure 40 below shows the pictorial representation of the PPTF Algorithm, while the succeeding table highlights the step-by-step algorithm workflow.

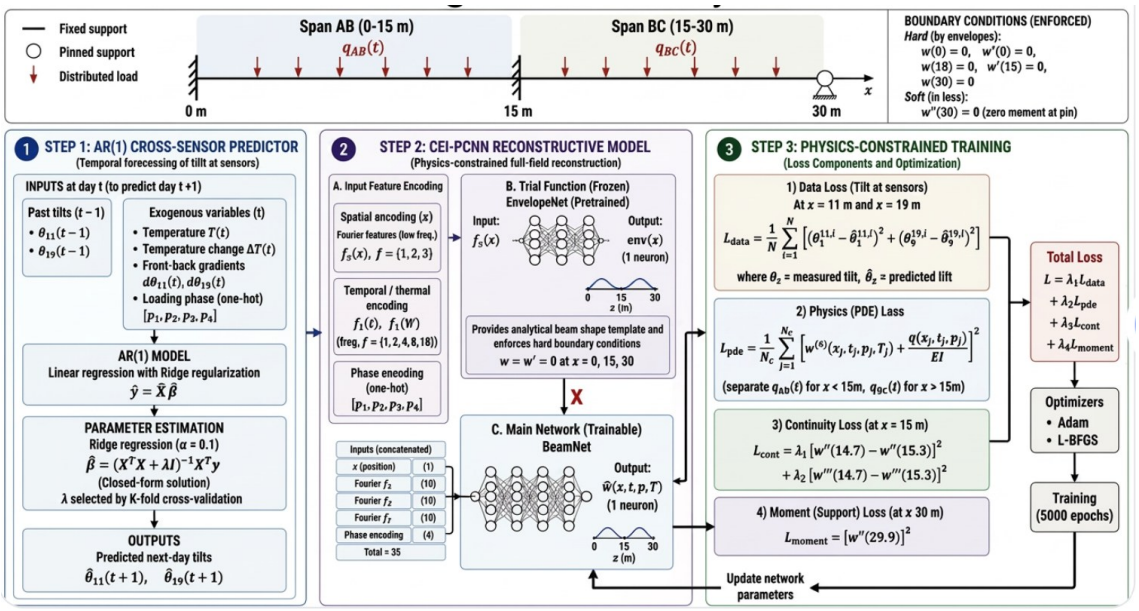


Figure 40: AR-PCNN Hybrid

AR-PCNN Hybrid Pipeline

Input: $s_{11}(t-1)$, $s_{19}(t-1)$ (yesterday's tilt)
$T(t)$, $dT_{11}(t)$, $dT_{19}(t)$, phase(t) (today's environmental data)
Pre-trained AR model, Pre-trained PCNN (EnvelopeNet)
Output: $\hat{s}_{11}(t)$, $\hat{s}_{19}(t)$ (predicted tilt at sensors)
$\theta(x,t)$, $v(x,t)$ $\forall x \in [0,30]$m (full-field reconstruction)
Prediction (AR Model)
$\hat{s}_{11}(t)$, $\hat{s}_{19}(t) \leftarrow AR(s_{11}(t-1), s_{19}(t-1), T, dT, phase)$
Reconstruction (PCNN EnvelopeNet)
$\theta(x,t)$, $v(x,t) \leftarrow PCNN(\hat{s}_{11}(t), \hat{s}_{19}(t), T, phase)$
Validation (test period days 214-253)
$\varepsilon\_sensor = \hat{s}(t) - s\_measured(t)$
$\varepsilon\_field = \theta\_PCNN(x,t) - \theta\_FEM(x,t)$ at $x \in \{5,10,20,25\}$m

APPENDIX 7 FEM Algorithm

Finite Element Method

Input: $s_{11}(t)$, $s_{19}(t) \in \mathbb{R}^N$ (daily tilt, N=253 days), phase(t) ∈ {1,2,3,4}, T(t) $\in \mathbb{R}^N$, L, w_grav
Output: UDLs: w_T, w_Tr, w_V3, w_V4 (kN/m, per span), θ(x,t), v(x,t) ∀x∈[0,30]m, t=1..N
Fe Unit Response
r_grav ← solve_beam(w_grav) ; r_unit ← solve_beam(-1.0)
s_g = get_slope(r_grav, {11,19}); s_u = get_slope(r_unit, {11,19})
Phase-By-Phase Discovery [mean matching per phase]
For p = 1..4:
D_p = {t : phase(t)=p}; $\bar{m}(p)$ = mean(measured_slope over D_p)
Phase 1: w_T = ($\bar{m}(1)$ - s_g) / s_u ; Phase 2: w_Tr = ($\bar{m}(2)$ - e(1)) / (-s_u)
Phase 3: w_V3 = ($\bar{m}(3)$ - e(2)) / (-s_u) ; Phase 4: w_V4 = ($\bar{m}(4)$ - e(3)) / (-s_u)
Update expected: e(p) = e(p-1) + w_new · (±s_u)
Accumulate dead: dead[p] = dead[p-1] + downward_loads
Per-Observation Attribution
For each t:
e(t) = s_g + Σ[phase-appropriate discovered loads × sensitivities]
residual(t) = measured(t) - e(t)
Δq(t) = residual(t) / (-s_u); net_UDL(t) = dead[phase(t)] + Δq(t)
Forward Reconstruction
For each x ∈ {0, 0.5, ..., 30.0}m, each t:
up_eq = w_grav - net_UDL(t, appropriate_span); θ(x,t) = -(θ_grav(x) + up_eq · θ_unit(x)) × 1000
v(x,t) = -(v_grav(x) + up_eq · v_unit(x)) × 1000
Validation
ε = max|θ_reconstructed(sensor) - measured| → should be 0 by construction

APPENDIX 8 Metric Table of all Models

This section compares the metrics of the four models: SPTF, PPTF, PETF, and AR-PCNN. The accuracy metrics are L^2, R^2, RMSE, and MAE. The physics metrics are boundary conditions violations, continuity condition violations, and PDE residual.

Pos. (m)	*Response*	*Model*	*R^2 ↑*	*Rel-L2 ↓*	*RMSE ↓*	*MAE ↓*
5	Slope	SPTF	0.8979	0.0916	0.0703	0.0584
		PPTF	-0.6324	0.3661	0.2809	0.2786
		PETF	0.6217	0.1762	0.1352	0.1305
		AR-PCNN	**0.9116**	**0.0852**	**0.0654**	**0.0562**
	Deflection	SPTF	0.7482	0.1438	0.5516	**0.4629**
		PPTF	-2.3362	0.5233	2.0075	1.9946
		PETF	0.7328	0.1481	0.5681	0.5331
		AR-PCNN	**0.7858**	**0.1326**	**0.5086**	0.4661
10	Slope	**SPTF**	**0.7465**	**0.1443**	**0.1107**	**0.0930**
		PPTF	-1.5919	0.4613	0.3539	0.3511
		PETF	0.6703	0.1645	0.1262	0.1186
		AR-PCNN	0.6639	0.1661	0.1274	0.1200
	Deflection	**SPTF**	**0.8576**	**0.1081**	**0.4148**	**0.3509**
		PPTF	-0.2657	0.3224	1.2365	1.2208
		PETF	0.7368	0.1470	0.5639	0.5308
		AR-PCNN	0.7308	0.1487	0.5702	0.5398
20	Slope	SPTF	-16.6211	0.3307	0.6926	0.6914
		PPTF	0.0815	0.0755	0.1581	0.1506
		PETF	**0.4302**	**0.0595**	**0.1245**	**0.1153**
		AR-PCNN	0.1575	0.0723	0.1514	0.1445
	Deflection	SPTF	-4.9383	0.1920	1.6555	1.6497
		PPTF	0.2391	0.0687	0.5926	0.5596
		PETF	0.3997	0.0610	0.5264	0.4969
		AR-PCNN	**0.4288**	**0.0595**	**0.5135**	**0.4840**
25	Slope	SPTF	-1192.0000	2.7211	2.6816	2.6795
		PPTF	-7.2013	0.2256	0.2223	0.2198
		PETF	-0.3691	0.0922	0.0908	0.0742
		AR-PCNN	**0.0700**	**0.0760**	**0.0749**	**0.0636**
	Deflection	SPTF	-91.7546	0.7587	9.3469	9.3377
		PPTF	-0.0441	0.0805	0.9917	0.9478
		PETF	**0.7677**	**0.0380**	**0.4678**	**0.3796**
		AR-PCNN	0.4355	0.0592	0.7292	0.6613

Physics Metric	*SPTF*	*PPTF*	*PETF*	*AR-PCNN*	*Best*
W''(30) RMS	5.44E−03	1.33E−04	8.36E−05	7.73E−05	AR-PCNN
Moment Continuity RMS	5.89E−05	4.69E−04	1.40E−04	1.41E−04	SPTF
Shear Continuity RMS	1.60E−05	6.92E−04	3.81E−04	3.67E−04	SPTF
PDE Residual AB RMS	3.39E−05	1.15E−04	2.07E−04	1.36E−04	SPTF
PDE Residual BC RMS	1.14E−04	3.57E−05	1.23E−04	4.91E−04	PPTF

APPENDIX 9: Ablation Analysis (Fourier Encoding vs Raw Inputs)

This section presents an ablation study of the Fourier-transformed features. The purpose is to ascertain whether it is necessary to transform the spatial input x, the time t, and the temperature T, before feeding them to the NN, or whether it is best to use them in their raw state. The ablation is made on SPTF, given that it is the simplest among all the PCNN architectures.

SPTF 1 (Raw) Input: (x, t_norm, T_norm, phase loading)

SPTF 2 (Fourier) Input: (x, t_fourier, T_fourier, phase loading)

The model named *"Fourier"* is simply an SPTF that uses Fourier-transformed features. The measured features are initially normalized and then transformed with Fourier cyclic encoding. The cyclic encoding transforms one input feature on five frequencies of sin and cos, resulting in 10 new features per input feature. In total, the input that goes in this model is 25 (i.e., 1x, 10 t_fourier, 10 T_fourier, and 4 one-hot encoded phase loading).

On the other hand, the model named *"Raw"* is an SPTF that uses the normalized features. These features are not transformed with Fourier cyclic encoding. The total features fed to the model is 7 (i.e 1x, 1 t_norm, 1 T_norm, and 4 one-hot encoded phase loading).

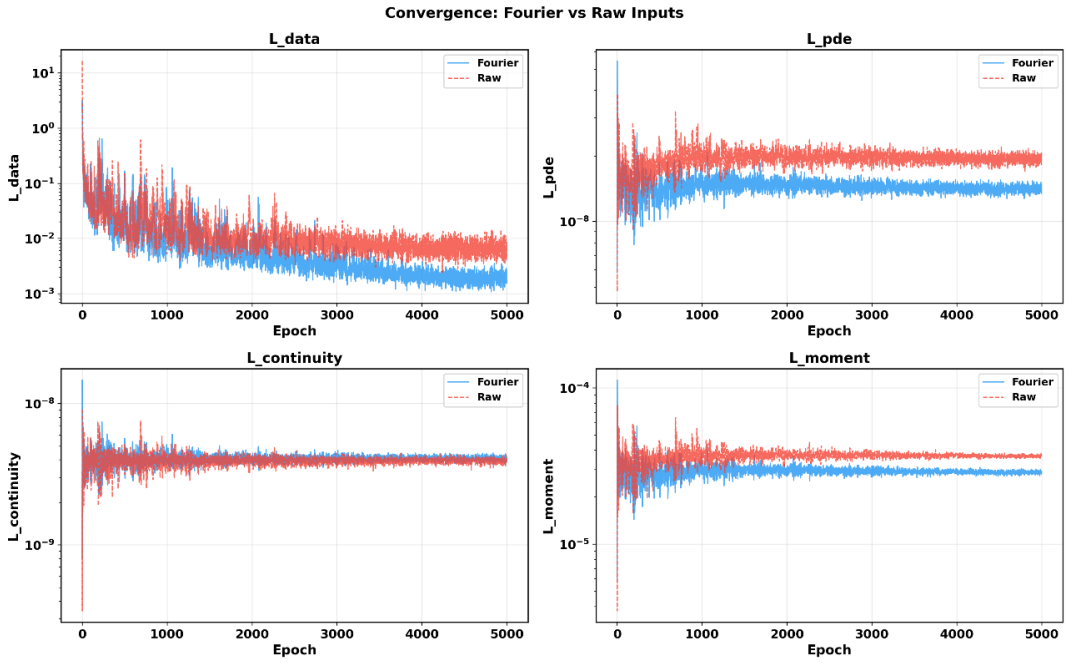


Figure 41: Ablation Convergence Plot

As shown in Figure 41, the SPTF with Fourier input features converges better than the raw input features. The data loss, PDE loss, and boundary condition loss (moment loss) for the Fourier-transformed features are clearly lower than those for the raw feature SPTF. This ascertains that transforming the time t and temperature T truly helped the model to learn some hidden patterns in the data. Additionally, the sensor fit of the SPTF with the Fourier-transformed features, as represented in Figure 42, has better correlation and less variance with the measured value in comparison with that of raw features.

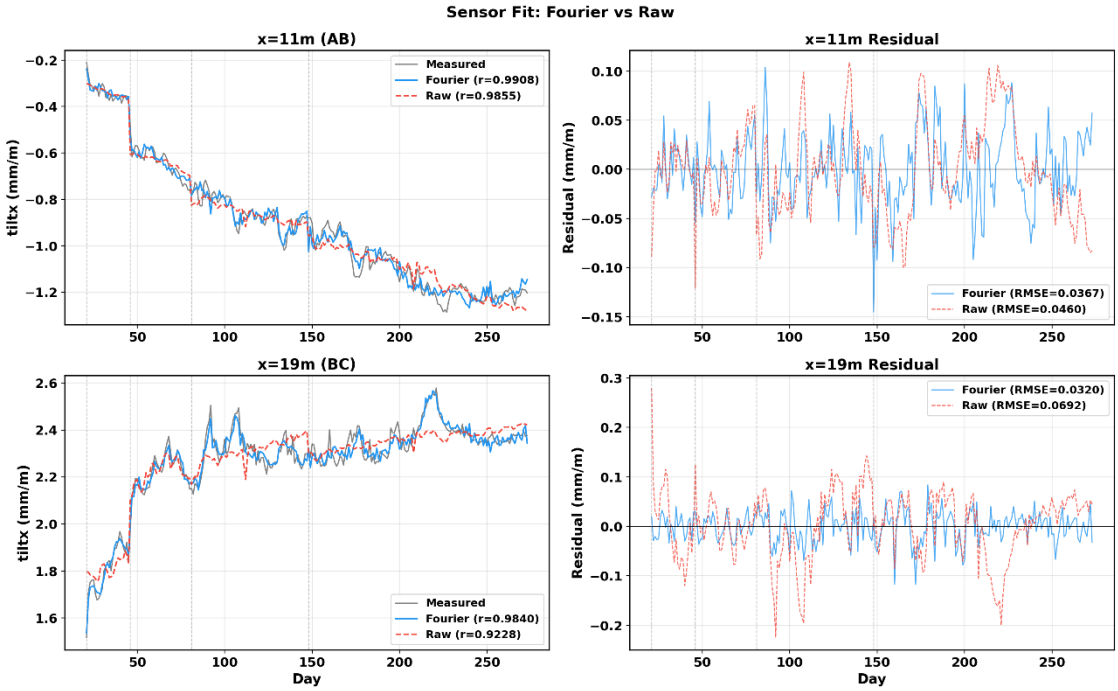


Figure 42: Ablation Sensor Fit

The metric comparison at two validation positions is presented in the table below for the ablation study

Position	*Response*	*Model*	*R^2 ↑*	*Rel-L2 ↓*	*RMSE ↓*	*MAE ↓*
x = 5 m	**Slope**	Fourier	0.9161	0.0830	0.0637	0.0533
		Raw	0.7309	0.1486	0.1140	0.0989
	Deflection	Fourier	0.7869	0.1323	0.5073	0.4220
		Raw	0.4137	0.2194	0.8416	0.7258
x = 20 m	**Slope**	Fourier	−16.3373	0.3280	0.6870	0.6856
		Raw	−18.9274	0.3517	0.7365	0.7333
	Deflection	Fourier	−4.9272	0.1918	1.6540	1.6474
		Raw	−5.3860	0.1991	1.7168	1.6953

In the two validation positions (x=5m and x=20m), the PCNN with Fourier-transformed features has better performance than the PCNN with the raw features.