DISSERTATION

Interpretable and Physics-Informed Machine Learning for Energy in Buildings

Author:
Jonathan Prigmore

Supervisor:
Dr Massimiliano Manfren

Department of Engineering
Aeronautics and Astronautics

Date: 02/05/2025

This report is submitted in partial fulfillment of the requirements for the Aeronautics and Astronautics, Faculty of Engineering and Physical Sciences, University of Southampton

Word Count: 9781

Declaration of Authorship

I, Jonathan Prigmore, declare that this dissertation and the work presented in it are my own, generated by me as the result of my own original research. I confirm that:

• This work was done wholly whilst in candidature for a degree at this University;

• Where any part of this thesis has previously been submitted for any other qualification at this University or any other institution, this has been clearly stated;

• Where I have consulted the published work of others, this is always clearly attributed;

• Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work;

• I have acknowledged all main sources of help;

• Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself;

• None of this work has been published before submission.

Acknowledgements

I thank my supervisor Dr Massimiliano Manfren for the opportunity to work on this topic, his unwavering support and technical expertise in this field. I also thank my parents and family for helping me to get this far.

Abstract

This report investigates the development of an interpretable and physics-informed framework that combines Resistance Capacitance (RC) networks with data-driven Machine Learning (ML) techniques for predicting thermal behaviour in buildings. This approach supports objectives in the Building Energy Modelling (BEM) sector, where both interpretability and predictive accuracy allow for fine-tuning of control systems. There are several innovations within the methodology of this report. Firstly, generalisability across RC network topologies makes it adaptable to models beyond the 5R1C configuration. Secondly, the reformulation of the applied 5R1C network from ISO 13790:2008 to the methodology, includes nodal manipulation, solar gain interaction terms and Boolean-based switching for ventilation schedules. The model was validated under free-floating conditions using ASHRAE 140 benchmarks. Time series data generated from the validated RC network were used to train and evaluate ML regression algorithms, specifically XGBoost and neural networks. Results indicate that XGBoost offered more accuracy and greater interpretability, making it a strong candidate for future physics-informed ML integration. The hybrid framework is scalable for BEM, supporting performance evaluation, retrofit analysis, and control optimisation. Ultimately, it aims to bridge the gap between data-driven prediction and thermodynamic consistency.

Contents

1 Introduction 7
1.1 Aim and Objectives 7

2 Literature Review and Background 9
2.1 Introduction to BEM 9
2.2 Historical Context and Advancements 9
2.3 Resistance Capacitance (RC) Thermal Network Models 10
2.3.1 Limitations 11
2.4 Physics-Informed Machine Learning (PIML) 11
2.4.1 Overview of PIML 11
2.4.2 Applications in BEM 12
2.5 Summary of Literature Review 13

3 Methodology 15
3.1 General Methodology for Solving RC Models 15
3.1.1 Calculation Steps 16
3.2 Assembled 5R1C ISO 13790:2008 Dynamic Network 19
3.2.1 Innovations from ISO 13790:2008 21
3.2.2 Model Calculation Steps 23

4 Results and Discussion 25
4.1 Physics-Based Grey-Box Model 25
4.1.1 Validation Against ASHRAE 140 Benchmarks 26
4.1.2 Sensitivity Analysis 28
4.1.3 Time-Series Data 29
4.2 Data-Driven Machine Learning 31
4.2.1 Uncertainty Visualisation Techniques in Regression Models 42

5 Conclusions and Further Work 48

6 Appendix A – Additional Equations 50

7 Appendix B – Spreadsheets 51

8 Appendix C – Code 57

9 References 67

Nomenclature

Symbols and Abbreviations

A	= Area
C	= Thermal capacitance
f	= Control factor
H	= Heat transfer coefficient
P	= Radiation split coefficient
R	= Thermal resistance
$\boldsymbol{R^2}$	= The coefficient of determination
T	= Temperature (general)
$\boldsymbol{\Delta\tau}$	=Time-step size
$\boldsymbol{\phi}$	=Heating and/or cooling power
θ	=Temperature
ASHRAE	= American Society of Heating, Refrigerating and Air-Conditioning Engineers
BEM	= Building Energy Modelling
BS	= British Standard
FF	= Free-Floating
HVAC	= Heating, Ventilation and Air Conditioning
IoT	= Internet of Things
ISO	= International Organization for Standardization
MAE	= Mean Absolute Error
ML	= Machine Learning
NN	= Neural Networks
PIML	= Physics-Informed Machine Learning

Subscripts

0	= Base; reference
10	= Reference or comparison condition

a = Refers to air node

conv = Convective component

e = External, exterior, envelope

HC = Heating and/or Cooling

int,C,set = Internal cooling set-point temperature

int,H,set = Internal heating set-point temperature

is = Conductance term

m = Refers to mass node

max = Maximum (heating or cooling capacity)

ms = Mass and surface interaction

nd = Need

occ = Occupied condition

op = Opaque

rad = Radiative component

rs, rm = Surface and mass radiation coefficients

rsd, rmd = Surface/mass solar gain distribution coefficients

s = Internal surfaces

set = Set-point temperature (heating or cooling)

set = set-point

sol = Solar (heat gains)

sup = Supply of air node

tr = Transmission (heat transfer)

un = Adjusted or floating condition

unocc = Unoccupied condition

ve = Ventilation

ws = Wall/surface interaction

1 Introduction

With the need for sustainable energy management and the complexity of modern buildings, Building Energy Modelling (BEM) has become a useful tool in the optimisation of a building's thermal performance [1]. Traditional BEM standards such as ISO 13790 and ISO 13792 employ entirely physics-based simulations. Whilst this approach offers an important theoretical foundation, it lacks the scalability and real-world application that the hybrid method, proposed in this study, aims to provide.

Recently, advancements in data-driven techniques with machine learning (ML) have gained popularity in the building sector. This method possesses important predictive capability but is deficient in transparency. This highlights an important gap which a hybrid approach aims to solve, combining the interpretability of the physical knowledge from physics-based simulations with the predictive abilities of ML. Here, interpretability represents the ability for humans to predict how the model output will deviate in response to a change in input data or algorithmic parameters, useful for retrofitting analysis. The presented framework in this report is a physics-informed simulation where the output data is used to train and assess ML algorithms. Through an enhanced standard Resistance-Capacitance (RC) network, the model's data is fed into ML regression models, and the proposed framework has predictive accuracy, while maintaining interpretability.

1.1 Aim and Objectives

The aim of this study is to simulate and predict a building's thermal behaviour with a novel hybrid modelling approach combining both RC networks with data-driven ML. This allows exploitation of simultaneous merits of both traditional simulation and modern data science approaches. To ensure this aim is achieved the following objectives of the report are:

1. To develop an adaptable methodology that can be applied to multiple building topologies and operational scenarios, making it suitable for larger-scale applications
2. Reformulate standardised and consolidated 5R1C (5 Resistances, 1 Capacitance) network model [2], testing against ASHRAE 140 (BESTEST) simulation cases [3]
3. Simulate this across cases of various thermal mass and ventilation schedules in the free-floating condition
4. Analyse the sensitivity of the RC model output

5. Use simulated data to test the predictive performance of data-driven machine learning methods, such as tree-based models with neural networks. This can be further developed to a physics-informed degree by leveraging RC model behaviour as a constraint
6. Analyse the uncertainty of ML prediction evaluation methods such as standard deviation and percentile banding. This reinforces confidence in predictive performance and offers a level of interpretability through visualisation

The objectives outline the goal of achieving a physics-consistent modelling methodology that aids the ever-evolving field of energy-efficient building design.

2 Literature Review and Background

2.1 Introduction to BEM

Buildings represent more than a third of global energy consumption and energy-related emissions [4-6]. The goal of achieving net zero carbon emissions by 2050 outlined in the Paris Agreement [7,8], showcases the need for energy efficiency in buildings. Consistent with this rising demand, BEM has emerged as an important process for designing energy-efficient systems [9]. In fact, the United Nations Sustainability Development Goals for 2030 dictate that the improvement of building energy performance has the potential to impact future global energy and emissions significantly [10].

BEM is a computational process designed for simulating energy and thermal dynamics. It is created through incorporation of architectural design, material properties and environmental factors. Through simulation, BEM predicts the energy usage under various operational and environmental conditions. This allows the identification of inefficiencies and prompts the optimisation of systems such as Heating, Ventilation and Air Conditioning (HVAC) [11]. The fine-tuning of the thermal dynamics involved in HVAC designs ensures maximum user comfort while minimising energy consumption, facilitating future retrofitting processes and optimised building design. The BEM process can be scaled to predict energy demand accurately over larger sets of infrastructure [12]. With these capabilities, BEM is essential for achieving net zero energy goals, meeting global climate commitments.

2.2 Historical Context and Advancements

BEM has to evolve its techniques in order to meet the demand for energy efficiency. The simulation methods are primarily split into two different approaches physics-based and data-driven.

Physics-based models incorporate finite element or volume methods and a system of differential equations to model the energy usage [2]. The equations can be modelled with computational software such as EnergyPlus which handles dynamic thermal simulation [13]. The simulation engines are based on thermal heat balance equations. Whilst this approach accounts for the intricate interactions between the building system, it requires extensive input data. These inputs are hard to obtain and are therefore replaced by standard assumptions meaning that the model is only validated once the building has been built. Additionally, this software is time-consuming and requires expertise. To lower this complexity, a milestone in BEM was the use of simplified lumped-parameter models, such as Resistance Capacitance (RC) networks. These models are a physics-

based approach capable of striking a balance between simplicity and accuracy, enabling engineers to calculate thermal behaviour without extensive, detailed simulation.

In recent years, the BEM industry has moved towards data-driven models, particularly those using Machine Learning (ML) methods [14]. These ML approaches are trained on historical data and real-time data from Internet of Things (IoT) sensors to predict and model energy performance [15]. Although ML models excel in the handling of large data sets, they are limited to the scope of the data upon which they were trained. An implication is that they struggle to predict data outside of what they know [16]. The principle types of machine learning models that are widely used in BEM are decision trees and Neural Networks (NN) [17-19]. While these are commonly known as "black-box" models they can be embedded with physical knowledge forming a grey-box model, a structure combining theory and data.

Addressing the challenges of both physics-based and data-driven methods, hybrid grey-box models have emerged as a modern solution [20,21]. In this report there is a combination of both the strengths of the physics-based model RC network and the data-driven techniques in ML. The physical understanding of the governing equations allows the output to be both interpretable and validated. Where interpretability represents the ability for a human to understand the rationale behind the logic [14], this synergy allows for improved accuracy and faster computational time.

2.3 Resistance Capacitance (RC) Thermal Network Models

An RC network uses an electric circuit to represent the energy balance in a building. It is a simplified method that models the heat transfer through materials with resistance (R) and the thermal capacitance (C), representing the ability to store heat. Ultimately, it is a dynamical system that models the thermal heat transfer over time. This report presents a methodology applicable to multiple RC networks, focusing specifically on 5R1C composed of five resistances and a single capacitance. The network and specific standard used in this study is validated in studies across large scale [22-24]. Ogunsola and Song [25] detail the assumptions in RC networks:

- The indoor air temperature is assumed to be uniform throughout the zone
- Windows are modelled as purely resistive elements, ignoring their thermal mass
- Long-wave radiation exchanges between internal surfaces are ignored
- Solar transmittance is zero for opaque components
- Convective internal loads are constant or change gradually over time
- Specific boundary conditions are applied to account for heat transfer through floors and ceilings
- Internal water vapour generation and heat transfer effects are neglected

The linear equations governing the RC thermal network can be modelled as a state space equation, describing the system using first-order differential equations [22]. This allows the prediction of energy demand under different conditions The models are less time-consuming than detailed simulation tools such as EnergyPlus [13]. Additionally, its simplicity allows quick integration into building control systems.

2.3.1 Limitations

While RC networks boast quick computational time and accurate predictions, there are some theoretical limitations [25]:

- Due to the simplified assumptions, RC models may not work with buildings with diverse topologies such as irregular geometries.
- RC models struggle to accurately represent energy exchanges between the return air and supply air
- The complexity of the model increases significantly with the number of building zones. With these greater zones, model order reduction techniques are essential.
- Variations in wind velocity result in non-uniform convection coefficients on external walls. Improved external flow dynamics help amend this.
- Non-uniform air distribution can lead to uneven convective heat transfer. This can be mitigated with the use of computational fluid dynamics (CFD)

2.4 Physics-Informed Machine Learning (PIML)

2.4.1 Overview of PIML

PIML is a hybrid modelling approach that combines physical knowledge into ML frameworks. Within the learning process, it embeds governing equations and domain-specific knowledge reducing the dependency on large data sets like in data-driven methods. The following literature outlines the key features of PIML:

1. Data:

Raissi *et al.* [26] display that PIML models, with their physical constraints, can calculate partial differential equations with a limited amount of data.

2. Interpretability:

Karniadakis *et al.* [27] showcased that since the calculations are from known principles, the authors argue that PIML allows for a better understanding of the model's decision-making process

3. Physically accurate:

Hao *et al.* [28] highlight how the physical constraints mean that the models are guided towards solutions that are physically plausible.

2.4.2 Applications in BEM

An application of PIML in BEM is the use of physics-informed inputs. This incorporates simulation data from physics-based models such as EnergyPlus and RC networks into data-driven models. The additional physical insight embedded in the simulation data enhances the performance of the ML models. Chen *et al.* [29] showcased that the coefficient of determination increased by 0.18 in a commercial building. In some cases, domain adaptation techniques are employed to address differences between observed data and simulation data, ensuring real-world applicability of the model [30]. This facilitates predicted energy usage with limited real-world data and enhances prediction accuracy in BEM [31].

PIML can also restructure the architecture of ML models used in BEM. This is achieved with the incorporation of physical knowledge into the architecture of data-driven models such as neural networks. For an accurate model the behaviour must reflect real-world behaviour, redesigning the structure to reflect physical constraints allows this. Bünning *et al.* [32] adopted this technique in neural networks, experimenting on real buildings and showcasing the benefits of PIML compared to ML. This study discovered that PIML had lower prediction errors and required less training data to achieve a good performance. While both methods effectively reduce energy consumption, PIML was superiorly efficient which is fundamental in real-time control scenarios.

An additional application of PIML in BEM is the use of physics-informed loss functions. Data-driven models optimise their performance based on the error between predicted and actual outputs. Where PIML differs is with the introduction of physical laws as regulation into the loss function, correcting deviations from the constraints. This means that results are in accordance with physical laws while fitting the data. Wang *et al.* [33] reconstructed the loss function by integrating a 2R2C model into the neural network. This study demonstrated that PIML outperformed the data-driven neural network, achieving a lower prediction error. The accuracy of results was dependent on the amount of physical knowledge incorporated, emphasising the importance of accurately reflecting real-world behaviour.

An innovative approach of PIML in BEM is the introduction of hybrid models, combining the strengths of physics-based and data-driven models. These are typically grey-box and

EnergyPlus models integrating their physics-based simulation components with ML to predict energy consumption. Hybrid models combining physics-based RC networks and EnergyPlus models with machine learning techniques are fully elucidated in literature. For example, Ma *et al.* [34] demonstrated significant improvements in predicting thermal loads, achieving better accuracy compared to traditional models from 40 to 90% [34]. Although this approach is relatively novel, it shows promise in collaborating the physical interpretability and predictive performance of ML, outlining the path for future developments in BEM.

However, there are several challenges surrounding the progression from data-driven ML to PIML. While data-driven methods such as NN and decision trees can embed physical constraints into their architecture, they are limited to the amount of data encoded, the more physical constraints the better. For example, lack of knowledge of physical laws, such as the conservation of energy, means the model may generalise dynamic behaviour. This is often counteracted using physics-informed loss functions, but this requires significant expertise and increases the model complexity. Whereas physics-based models alone are relatively simple and yield an accurate result. Consequently, the industrial progression towards PIML is not straightforward and requires a systematic review of the physics-based simulations and the correct ML model to advance. This involves the careful selection of training data for ML and suitable data science techniques to be chosen. The innovations in this report allow the model to act as an accurate representation of real thermodynamic behaviour. Although the ML models do not possess physically embedded simulations, they are trained on data grounded in RC thermal networks.

2.5 Summary of Literature Review

It is clear that the available literature creates a strong foundation for the development of the methodology in this report. With the assumed trajectory of this industry, there is a clear demand for interpretable, accurate and efficient models in BEM. Addressing this, there are systems combining physics-based and ML approaches.

RC thermal networks, a physics-based approach, offer thermodynamically valid simulations for modelling building dynamics. This method is also currently employed in the British standard home energy model, indicating the relevance and reputability of simulations [35]. The formulation of the networks' linear equations into a state-space equation allows an analytically tractable result and seamless integration of control strategies. However, simplified assumptions of behaviour in the model mean the model is limited and requires adaptation for more complex topologies, which is explored in this methodology.

The literature on data-driven methods details the strong predictive capabilities of ML when trained on large data sets. However, it introduces potential issues of interpretability and

generalisation which can have a large impact on BEM results. This is emphasised by the drastic improvement of results in data-driven methods when physical knowledge was incorporated [34]. These developments in PIML embed the physical constraints, using physics-based simulation data as an input, adapt architecture and design loss functions to improve the realistic applications of ML.

Ultimately, this project draws on the positives of either method, constructing a novel hybrid grey-box modelling framework and assessing the results of data-driven ML trained on its output. The innovations of the 5R1C model (ISO 13790/13792) supports time-dependent dynamics, logical switching between coefficients and real-world application of simulations. Although physical constraints are not fully embedded within the ML models, the structure of data is informed by RC model simulations and their physical calculations. This outlines a base for PIML where a further advancement incorporates physical knowledge into the framework itself.

The following table outlines the literature concepts and their effect on the work produced in this study:

Table 1: Applying Literature Review to the Study

Literature Concept	**Application in this Study**
RC Networks (5R1C model, ISO 13790/13792)	Adapted for time-dependent dynamics and integer variables to switch between time-dependent coefficients in simulations
Hybrid Grey-Box Modelling	RC simulation outputs used to generate data for training data-driven ML models
PIML Frameworks (architecture, loss, inputs)	Aids the conceptual guidance of PIML frameworks based on this study
Interpretability in ML	Helps decipher which ML method is more effective, for a traceable model structure and rule-based predictions in thermal modelling
ML Scalability and Adaptability	Supports flexible prediction for different case configurations, ventilation patterns, and masses

The integrative methodology, advised through this literature review, acts as a foundation for future work towards scalable, accurate and interpretable predictive tools in BEM.

3 Methodology

The methodology presented in this report contains two innovative elements. Firstly, the methodology applies to multiple RC models, with the correct set of governing linear equations. This method is applied to a reformulation of 5R1C from ISO 13790:2008 standard to validate the generalised methodology for further application. To ensure this validation, the output is tested against ASHRAE 140 benchmarks. Secondly, the reformulation of equations is an innovation from state-of-the-art standards. The distribution of solar and internal gains to nodes means the thermal network more explicitly accounts for solar interactions. Additionally, while it is equivalent to standards, introduced manipulation of certain coefficients avoids mathematical errors in certain situations.

3.1 General Methodology for Solving RC Models

Successfully modelling and simulating RC thermal networks is essential to understand the dynamic behaviour of buildings for energy management. The methodology presented uses matrix inversion to simplify the calculation of temperatures in the thermal model. This method remains flexible and can be applied to multiple RC network configurations like 5R1C (5 resistors, 1 capacitor) and 7R2C (7 resistors, 2 capacitors) commonly used in BEM. This matrix-inversion approach is reported in depth in a paper by Lundström *et al* [36] where it is shown to work on a large-scale case study. The key innovation of this approach is its ability to handle various configurations and conditions, without the need for a complete recalibration of the model for each case, making it highly scalable and efficient [12]. In general, the system of equations governing the RC network can be represented in matrix form:

$$A \times X = B \tag{1}$$

here, A is the coefficient matrix, representing the relationships between the resistances and capacitances in the network. X is the state vector containing the unknown values such as the node's temperatures and B is the vector of boundary conditions and previous time step values.

This system must be solved at every time step. One of the most computationally intensive steps in solving the system of equations is the matrix inversion. This step is necessary to solve the state vector X at each time step:

$$X = A^{-1} \times B \tag{2}$$

By assuming matrix A has quasi-steady state variables, the matrix does not change between time steps. This allows the pre-inversion of the matrix, meaning that the inverse matrix A^{-1} is computed

only once, at the beginning of the simulation. The benefit of this is that even if the coefficients exhibit small oscillations, the same inverse matrix can be reused without needing to recompute it at each individual time step. This speeds up the simulation process considerably, especially for large scale or long-duration modelling where multiple time steps are required.

3.1.1 Calculation Steps

The algorithmic approach of simulations begins in the free-floating state, assuming no active heating or cooling is applied, allowing the determination of the initial temperature of the nodes. This state, without this additional temperature control, provides an initial baseline for the systems behaviour. This allows the algorithm to deduce whether or not heating or cooling is required based on the current indoor air temperature.

Boolean state variables are introduced for heating and cooling respectively, comparing the indoor temperature to set-point thresholds. This simplifies the computational process ensuring that only relevant calculations are made at each time step. If the current temperature is below the desired range, such that there is a need for heating, the heating variable is set to 1. For a need of cooling if the current temperature is above the upper limit the heating variable is set to 1. Within the acceptable range both variables, remain at 0 and no action is taken for the time step.

Following this, is the calculation of the required power to heat or cool to the set-point temperature. The algorithm verifies whether or not the heating or cooling system can provide the required power, if not the conditions are recalculated based on the maximum available power. The five possible situations after the internal temperatures are calculated are summarised below in Table 2:

Table 2: Conditional Required Heating and Cooling

Condition	Description	Outcome
$\boldsymbol{\theta_{air} < \theta_{int,H,set}}$	Heating is required but the building has insufficient heating power	The heating power is set to the maximum capacity of the heat pump
$\boldsymbol{\theta_{air} = \theta_{int,H,set}}$	Heating is required and the building has sufficient heating power	The calculated heating power is applied, which is lower than the maximum capacity of the heat pump
$\boldsymbol{\theta_{int,H,set} < \theta_{air} < \theta_{int,C,set}}$	No heating or cooling needed.	The building is in free-floating conditions and no additional heating or cooling is applied
$\boldsymbol{\theta_{air} = \theta_{int,C,set}}$	Cooling is required and the building has sufficient cooling power	The calculated cooling power is applied, which is lower than the maximum capacity of the cooling
$\boldsymbol{\theta_{air} > \theta_{int,C,set}}$	Cooling is required but the building has insufficient cooling power	The cooling power is set to the maximum capacity of the cooling

The piecewise function illustrated in Figure 1 based on EN ISO 13790:2008, calculates the heating or cooling power required depending on the temperature difference between the air and the set-points. The function operates based on the following rules. Firstly, the unrestricted heating or cooling required is calculated by:

$$\phi_{HC,nd,un} = \phi_{HC,nd,10} * \frac{\theta_{air,set} - \theta_{air,0}}{\theta_{air,10} - \theta_{air,0}} \quad (3)$$

where $\phi_{HC,nd,un}$is the heating or cooling power needed and $\phi_{HC,nd,10}$ is a scaling factor. This equation ensures that the heating power varies proportionally with the temperature difference.

Secondly, the boundary conditions which govern what heating or cooling power is required:

$$\phi_{HC,nd,un} = \begin{cases} 0, & \theta_{int,H,set} < \theta_{air} < \theta_{int,C,set} \\ \phi_{HC,nd,un}, & \phi_{C,max} < \phi_{HC,nd,un} < \phi_{H,max} \\ \phi_{H,max}, & \phi_{HC,nd,un} > \phi_{H,max} \\ \phi_{C,max}, & \phi_{HC,nd,un} < \phi_{C,max} \end{cases} \quad (4)$$

where $\phi_{H,max}$, $\phi_{C,max}$ represent the maximum possible heating or cooling power. The function has different calculation methods depending on where the current temperature is in comparison to the set-points for heating or cooling. The power is calculated based on the following conditions. If the current temperature θ_{air} is below the heating set-point $\theta_{int,H,set}$, then heating power is required, and the model adjusts the system to provide heating. If the system is in the free-floating state, with no active heating or cooling power, when the temperature is in between the set-points of heating and cooling. Lastly, if the power required exceeds the maximum power the system is limited to that value.

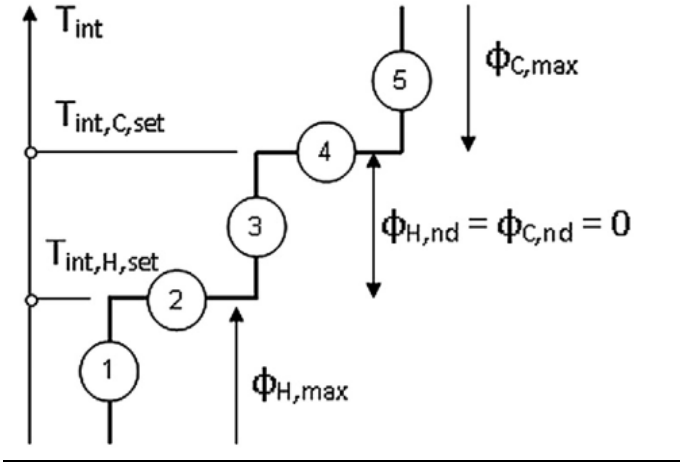

Figure 1: Heating or Cooling Power According to ISO13790:2008[2]

The piecewise function in Eq. (4) is an important part of the algorithm, ensuring heating and cooling power are correctly assigned based on the current temperature, while limiting it to the system's capabilities. Through calculation of the required power based on the temperature difference between set-points, this means that the algorithm is applicable to multiple scenarios. The maximum available power $\phi_{H,max}$ or $\phi_{C,max}$ in the algorithm ensures the system provides a controlled response and does not overdrive resources.

The following table summarises the generalised methodology presented for solving RC models applicable to a variety of RC thermal networks with the correct linear equations:

Table 3: Generalised Method of Solving RC Models

Step	Task	Description
1	Matrix setup	Define the system's coefficient matrix A, state vector X, and boundary vector B
2	Matrix inversion	Pre-invert the matrix A to save computation time during the simulation
3	Free-floating state	Check if heating or cooling is required based on temperature set-points
4	Power calculation	Calculate the required heating or cooling power using the equations provided
5	Capacity check	Compare the calculated power to the system's maximum capacity and adjust the power if necessary
6	Repeat	Update the system for each time step using the correct pre-inverted matrix

3.2 Assembled 5R1C ISO 13790:2008 Dynamic Network

While the ISO 13790:2008 is not the current standard ISO 52016-1:2017, it has been validated in a greater number of scientific works in comparison [37-41]. In this standard each physical element, such as the roof windows and walls, are represented as lumped parameters in 5R1C. The model used employs a simplistic hourly model for building thermal simulations, simplifying the complex interactions into tractable forms of mathematics. The energy balance calculations are governed by the parameters of resistances (R), thermal capacitance (C), and the heat transfer coefficient (H) which describes the flow of heat between the different parameters. Figure 2 shows two equivalent representations of the 5R1C thermal network for a building zone.

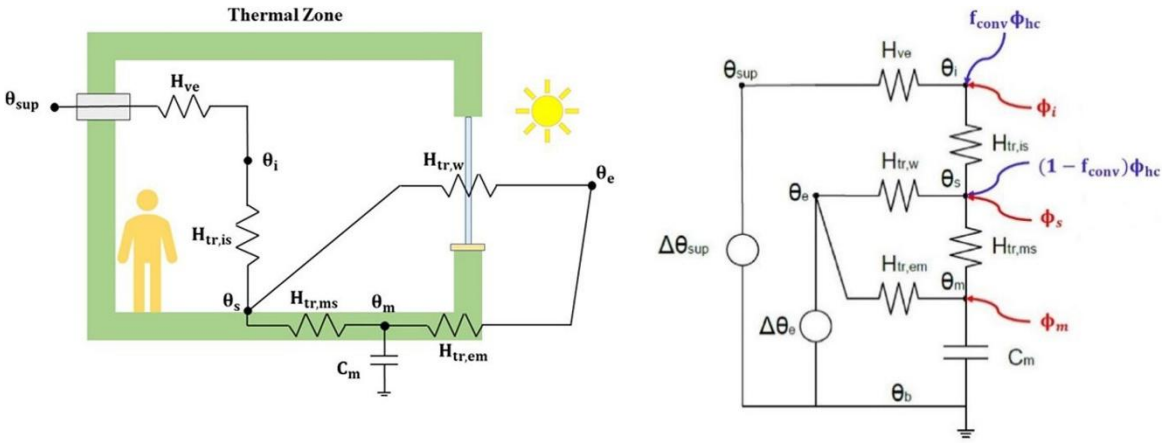

Figure 2: Illustrations of ISO13790:2008 5R1C Thermal Network [41]

In the thermal model the system is divided into respective nodes such as the air, surface and mass nodes. These represent the physical spaces in the building, where the lumped parameters determine the heat flow around them. Through material and surface properties the heat transfer coefficients are calculated. Firstly, the heat transfer from infiltration or external environment to the indoor air is determined using:

$$H_{tr,is} = h_{is} \times A_t \tag{5}$$

where h_{is} is the infiltration heat transfer coefficient. A_t is the total surface area of the building through which heat is exchanged with the environment. For the heat transfer through walls and surfaces, the heat transfer coefficient can be calculated using:

$$H_{tr,ws} = h_{ws} \times (A_t - A_m) \tag{6}$$

where h_{ws} is the heat transfer coefficient for walls and surfaces. A_m is the area of the mass node, the material that absorbs and releases heat. For the heat transfer between the mass and the air node is given by:

$$H_{tr,ms} = h_{ms} \times A_m \tag{7}$$

where h_{ms} is the heat transfer coefficient between the mass and air node. Lastly, for the total heat transfer between the external environment and the mass surface node is determined using the formula:

$$H_{tr,em} = \frac{1}{\frac{1}{H_{tr,op}} - \frac{1}{H_{tr,ms}}} = \frac{H_{tr,op}H_{tr,ms}}{H_{tr,ms} - H_{tr,op}} \tag{8}$$

where $H_{tr,op}$ is the heat transfer coefficient for external heat exchange (operation heat transfer coefficient).

3.2.1 Innovations from ISO 13790:2008

An issue that may arise in the original formulations of the 5R1C model according to ISO 13790:2008 is the potential for zero denominators, particularly in Eq.(8). This issue may occur when underlying coefficients in the denominator are zero. To avoid this, alternative formulas are used allowing the model to be applied under all conditions, including those with ventilation. Here the heat transfer coefficients are split up into three nodes and this can be proven to be equivalent to ISO 13790:2008 Eq.(5-8). The heat transfer coefficient for Node 1 is calculated with:

$$H_{tr,1} = \frac{1}{\frac{1}{H_{ve}} + \frac{1}{H_{tr,is}}} = \frac{H_{ve}H_{tr,is}}{H_{ve} + H_{tr,is}} \tag{9}$$

where $H_{ve}$is the heat transfer coefficient of ventilation. The heat transfer coefficient for Node 2 is calculated by:

$$H_{tr,2} = H_{tr,1} + H_{tr,w} \tag{10}$$

Finally the heat transfer coefficient for Node 3 is determined by:

$$H_{tr,3} = \frac{1}{\frac{1}{H_{tr,2}} + \frac{1}{H_{tr,ms}}} = \frac{H_{tr,2}H_{tr,ms}}{H_{tr,2} + H_{tr,ms}} \tag{11}$$

This develops a more refined and accurate thermal hourly model for 5R1C. Reformulation of the nodal gains and losses from the ISO 13790:2008 and ISO 13792:2012 standards [2,42], more specifically acknowledges the gains from convective and solar heat sources. This enhancement means the model has a greater understanding of the actual thermal heat transfers within the building.

In the original standards, the total heat gain to the air node (Φ_i) and surface node (Φ_s), essentially the convective and radiative gains, were calculated using these formulations:

$$a_{conv} = 0.5 \tag{12}$$

$$a_{conv} + a_{rad} = 1 \tag{13}$$

$$\Phi_i = a_{conv}\Phi_i \tag{14}$$

$$\Phi_s = \frac{A_m}{A_t}(a_{rad}\Phi_s + \Phi_{sol}) \tag{15}$$

$$\Phi_s = \left(1 - \frac{A_m}{A_t} - \frac{H_{tr,w}}{9.1A_t}\right)(0.5\Phi_s + \Phi_{sol}) \tag{16}$$

where a_{conv} is the convective coefficient, a_{rad} is the radiative coefficient, Φ_i is the internal heat gain to the air node, Φ_s is the solar gain on the surface node, and Φ_{sol} is the solar heat gain. The

physical meaning of the value of 9.1 in Eq.(16) represents a heat transfer coefficient correction factor derived from ISO 13790. This accounts for the convective heat transfer resistance between the air and the walls of the building.

The first reformulation of nodal gains and losses from Eq. (12-16) incorporates a coefficient of solar-air interaction. This means the thermal model now explicitly accounts for interactions between the solar radiation and the air node. The first modified formulations are as follows:

$$\Phi_i = a_{conv}\Phi_i + a_{sol-air}\Phi_{sol} \tag{17}$$

$$\Phi_s = \left(1 - \frac{A_m}{A_t} - \frac{H_{tr,w}}{9.1A_t}\right)(0.5\Phi_s + (1 - a_{sol-air})\Phi_{sol}) \tag{18}$$

where $a_{sol-air}$ has a value of 0.1, a technical standardisation [43].

Extending this adaptation from ISO 13790, there is the introduction of new coefficients and new nomenclature. These new formulations can be proven to be equal to ISO 13790, but they account for the interactions between the gains and the buildings properties more accurately. The adapted equations are as follows:

$$\Phi_s = P_{rs}(a_{rad}\Phi_s) + P_{rsd}(1 - a_{sol-air})\Phi_{sol} \tag{19}$$

$$\Phi_m = P_{rm}(a_{rad}\Phi_s) + P_{rmd}(1 - a_{sol-air})\Phi_{sol} \tag{20}$$

$$P_{rs} = \frac{A_t - A_m}{A_t}\left(1 - \frac{H_{tr,w}}{h_{ws}(A_t - A_m)}\right) \tag{21}$$

$$P_{rm} = \frac{A_m}{A_t} \tag{22}$$

$$P_{rsd} = \frac{A_t - A_m - A_w}{A_t - A_w}\left(1 - \frac{H_{tr,w}}{h_{ws}(A_t - A_m)}\right) \tag{23}$$

$$P_{rmd} = \frac{A_m}{A_t - A_w} \tag{24}$$

where P_{rs} accounts for the areas and thermal resistances of walls and windows. This value adjusts depending on the interactions between solar and internal heat gains. P_{rm} adjusts accordingly with the mass areas in the building. P_{rsd} and P_{rmd} adjust the solar gains and mass heat interactions based on the external environment and material properties

Together the innovations based on ISO 13790:2008 and ISO 13792:2012 play an important role in the fidelity and application of the hourly grey-box RC model developed in this study. The avoidance of zero errors improves the numerical stability of simulations allowing for broader applications of the model. The expanded nodal gains and losses give greater insight into the internal

and solar heat dynamics of particular concern under varying ventilation schedules. Furthermore, this makes the model more suitable for the integration of ML as it supports higher fidelity predictions.

3.2.2 Model Calculation Steps

For the 5R1C network the generalised methodology, previously outlined, is employed. Where in Eq.(1) A represents a 3×3 square matrix that holds the systems coefficients. Inputting the linear equations which govern the thermal dynamics of the 5R1C model into this equation gives:

$$\begin{Bmatrix} 1 + (\frac{H_{tr,em} + H_{tr,ms}}{\frac{C_m}{\Delta\tau}}) & -\frac{H_{tr,ms}}{\frac{C_m}{\Delta\tau}} & 0 \\ -H_{tr,ms} & H_{tr,ms} + H_{tr,w} + H_{tr,is} & -H_{tr,is} \\ 0 & -H_{tr,is} & H_{tr,is} + H_{ve} \end{Bmatrix} \times \begin{Bmatrix} \Theta_{m,i} \\ \Theta_{sup} \\ \Theta_{air} \end{Bmatrix} = \begin{Bmatrix} \Theta_{m,i-1} + \frac{H_{tr,ms}}{\frac{C_m}{\Delta\tau}}\Theta_e + \phi_{m,i} + f_m\phi_{HC.nd} \\ H_{tr,w}\Theta_e + \phi_{s,i} + f_s\phi_{HC,nd} \\ H_{ve}\Theta_e + \phi_a + f_a\phi_{HC,nd} \end{Bmatrix} \tag{25}$$

The matrix A remains constant between time steps, meaning that it does not change significantly. Therefore, the matrix inversion can be pre-computed at the start of the simulation:

$$\begin{Bmatrix} \Theta_{m,i} \\ \Theta_{sup} \\ \Theta_{air} \end{Bmatrix} = A^{-1} \times \begin{Bmatrix} \Theta_{m,i-1} + \frac{H_{tr,em}}{\frac{C_m}{\Delta\tau}}\Theta_e + (\frac{\phi_m + f_m\phi_{HC.nd}}{\frac{C_m}{\Delta\tau}}) \\ H_{tr,w}\Theta_e + \phi_{st} + f_s\phi_{HC,nd} \\ H_{ve}\Theta_e + \phi_{ia} + f_a\phi_{HC,nd} \end{Bmatrix} \tag{26}$$

The specific inverted matrix, applied to each time step, allows the calculation of the temperature values. The main concern in BEM is the value of indoor air temperature a direct indicator of thermal comfort. The adapted 5R1C thermal model employed creates a grey-box model producing hourly results of temperature from determined coefficients. Thermal simulations of the free-floating case, without any active heating or cooling, serve as a base for the model's validation. The model presented in this report has been tested on four different ASHRAE 140 (BESTEST) cases with varying ventilation schedules [3].

The varying schedules require adaptation to the framework determining the value of H_{ve} depending on the time step. This is emphasised in a free-floating scenario where ventilation is one of the few impactors of the building's thermal performance. The cases analysed incorporate two different ventilation schedules, occupied and unoccupied. For the occupied schedule the higher

ventilation rates are employed and the schedule value is set to 1. Whereas, for the unoccupied ventilation schedule lower ventilation rates are employed and the daily schedule value is set to 0. To seamlessly switch between heat transfer ventilation coefficients for simulations the dynamical calculation of Boolean switching facilitates this. Piecewise linearisation methods like this are frequently employed in engineering and control literature for hybrid dynamic systems [44]. The coefficient is calculated by:

$$H_{ve} = H_{ve,unocc} + \left(H_{ve,occ} - H_{ve,unocc}\right) \cdot Daily\ Schedule \tag{27}$$

where $H_{ve,unocc}$, $H_{ve,occ}$ are the unoccupied and occupied heat transfer coefficients respectively. The $Daily\ Schedule$ represents the binary variable, where 1 indicates ventilation is active and 0 when ventilation is reduced. This equation makes certain that depending on the schedule the specific heat ventilation coefficient is used. Resultantly, two unique inverted A matrices are applied at each time step depending on the ventilation schedule. Meanwhile, the B matrix is also updated dynamically, corresponding to the boundary condition. Substituting Eq.(27) into Eq.(25) for the A matrix gives:

$$A = \begin{Bmatrix} 1 + \left(\frac{H_{tr,em} + H_{tr,ms}}{\frac{C_m}{\Delta\tau}}\right) & -\frac{H_{tr,ms}}{\frac{C_m}{\Delta\tau}} & 0 \\ -H_{tr,ms} & H_{tr,ms} + H_{tr,w} + H_{tr,is} & -H_{tr,is} \\ 0 & -H_{tr,is} & H_{tr,is} + H_{ve,unocc} + \left(H_{ve,occ} - H_{ve,unocc}\right) \cdot Daily\ Schedule \end{Bmatrix} \tag{28}$$

This ensures that there is a swift interchange of equations between ventilation states without the need for extra calculation. This leaves two unique pre-inverted matrices for simulations, improving the computational efficiency when applied to each time step.

The results of the RC model simulations and matrix inversions are validated against standard benchmark cases. Here the indoor air temperature must fall within the expected range. The free-floating cases are tested against the standardised ranges from ASHRAE 140 and ISO 13790 benchmarks [2-3]. The model in this report is considered valid if the minimum, maximum and average indoor air temperature falls within the expected ranges. A pass or fail is given to each value based on the thresholds.

Although the free-floating model is only the base of the general methodology it provides insight into the model's accuracy. This condition highlights the effect of the thermal mass, envelope properties and ventilation interactions on the thermal model. If the model can accurately replicate thermal behaviour and pass against standards it means subsequent simulations including heating and cooling will behave accurately, thus validating this methodology.

4 Results and Discussion

The results and discussion section is split into two parts. Firstly the physics-based model, assessing the results produced. Secondly, the data-driven ML models which are trained on the final data from this analysis.

4.1 Physics-Based Grey-Box Model

As a foundational test of this methodology, a free-floating model is used. In the absence of heating and cooling, this represents the baseline thermal behaviour of buildings under varying environmental conditions. The model uses time-series data across a year at hourly intervals equating to 8,760 time steps. A short sample of this data for each case can be found in *Appendix B.* Fine hourly resolution encapsulates any delayed responses within the building mass, essential for assessing thermal comfort. This provides valuable insight into seasonal variability on thermal dynamics. Calculated using the RC grey-box modelling approach, each time step possesses a different energy balance. In ISO 13790 [2], The Crank-Nicolson method solves the dynamic problem of the RC network these equations are shown in *Appendix A.2*. The equations are reformulated with heating and cooling gains split to nodes providing an implicit solution scheme.

Blending physical modelling and empirical parameter calibration, this grey-box model is used. The fundamental coefficients of thermal resistances, capacitances, and gains are pre-calculated within an Excel spreadsheet using the adapted ISO standard 5R1C formulation. This allows the iterative calculation of nodal temperatures for each time step, producing data for three temperatures including indoor air temperature, surface temperature and mass temperature. Indoor air temperature serves as the main benchmark for the model's validation as it most directly represents thermal comfort, a main concern of residents. Surface and mass temperatures, while not used for validation, offer a deeper level of interpretability of the internal thermal storage and energy flux across the nodes.

Selected are four different ASHRAE 140 cases 600FF, 650FF, 900FF and 950FF [3]. These are chosen for their varying internal gains and broad range of thermal mass and ventilation occupancy. For instance, case 600FF is a lightweight construction whereas 900FF represents a structure of a heavier weight. Both 600FF and 900FF possess a standard ventilation schedule of occupancy, but 650FF and 950FF include reduced ventilation during unoccupied periods from 19:00 to 07:00, *Appendix B* shows this in detail. This added variation serves as an important indicator for how the model handles dynamic ventilation and is a further case for validation.

A Python-based solver extracts relevant determined verification coefficients from the grey-box model applying them externally to the 5R1C matrix formulation. This can be found in *Appendix C.1*. This constructs a relevant dynamic A and B matrix for every 8,760-time steps. Logically speaking, the Boolean switching variable can be used, however, the solver can identify the ventilation schedule and assign the correct matrices for that time step. This ensures computational efficiency as only two pre-inverted matrices are required.

The values of the indoor air temperature gathered from the solver across the year, are used to identify the minimum, maximum and average air temperatures. These values provide comprehensive insight into the performance of the building and are compared to the time series output of the grey-box model as an important step to validate the solver. This also reinforces consistency between the two independently computed outputs. Furthermore, the values are used to verify the output falls within ASHRAE 140 standards ensuring reliability and accuracy of the model under various cases.

4.1.1 Validation Against ASHRAE 140 Benchmarks

Indoor air temperature is selected as the validation metric for each of the four ASHRAE cases and are evaluated against pass/fail thresholds. These are determined by the ASHRAE 140 standard and represent the acceptable performance boundaries for the minimum, maximum and average indoor air temperatures over a calendar year [3].

Using the coefficients extracted from the grey-box model, the results from the Python-based solver are shown in Table 4. The thermal behaviour of each case is assessed based on its annual statistics. Table 4 showcases these values with the respective pass/fail ranges to determine their performance and compliance with the ASHRAE 140 requirements.

Table 4: Solver Results Benchmarked for Each Free-Floating Case Against ASHRAE 140 [3]

Case	Min (°C)	Min Range (°C)	Status	Max (°C)	Max Range (°C)	Status	Avg (°C)	Avg Range (°C)	Status
600FF	-18.95	-18.8 to -15.6	Fail	66.96	64.9 to 75.1	Pass	26.28	24.2 to 27.4	Pass
650FF	-23.15	-23.0 to -21.0	Fail	65.62	63.2 to 73.5	Pass	19.85	18.0 to 20.8	Pass
900FF	-6.78	-6.4 to -1.6	Fail	45.72	41.8 to 46.4	Pass	26.22	24.5 to 27.5	Pass
950FF	-20.42	-20.2 to -17.8	Fail	38.40	35.5 to 38.5	Pass	14.59	14.0 to 15.3	Pass

Evidently, Table 4 indicates that for all cases the values of maximum and average air temperature pass. However, all cases failed in the minimum value for indoor air. These are relatively fine margins of error, with most cases possessing as large as a 1% difference. Case 900FF has a margin of error of 6%, which is only 0.38 °C due to the magnitude of the value. There is greater percentage deviation from the pass range for the 600FF and 900FF cases, both of which employ a fully occupied ventilation schedule, compared to their counterparts that use reduced ventilation during unoccupied periods. This suggests that the model may be overestimating heat losses during this time. This behaviour displays the relationship between ventilation-related heat losses and the building's thermal characteristics. In the cases where the ventilation remains occupant for longer durations the building is subject to greater thermal flux, emphasising the accuracy of the model's representation of thermal inertia.

The consistent underperformance of the minimum value prompts an investigation into the properties of the model specifically concerning the thermal inertia and heat properties of the model, where this issue may have occurred. Since the coefficients define the thermal mass and heat transfer, the capturing of low temperatures relies on the accuracy of these values, particularly during nighttime and winter. Whilst the success of both calculated maximum and average temperatures boasts confidence in the reliability of the model, the failure of the minimum warrants further

scrutiny of the model's structure. The synergy of results between the python-solver and the grey-box model output allows the validation of the process of how values were calculated. The time series data across both platforms are identical reinforcing that the error does not arise from a mathematical flaw in either method, but the fundamental parameter assumptions. Thus, the stage is set for the subsequent sensitivity analysis on a particular coefficient and its influence on performance.

The analysis specifically focuses on variations in $C_m/\Delta\tau$, the ratio of thermal capacity and time step. This coefficient plays an important role in the ability to simulate thermal inertia and also has a great effect on the values of the matrices. One consequence of this value being lower than expected, is the reduced thermal buffering, resulting in faster cooling-down periods on winter nights. This contributes to the underestimated minimum indoor temperatures seen across all four free-floating cases. The coefficient has been chosen for adjustment due to its direct link to the dynamic thermal response of the model. Not only does this step mean the model complies with ASHRAE 140 but it also strengthens the model to more real-world scenarios beyond the four cases.

4.1.2 Sensitivity Analysis

Informed by the consistent failure to meet the ASHRAE 140 minimum temperature benchmark, a sensitivity analysis was performed on $C_m/\Delta\tau$. The refinement of this parameter is set to assess whether a moderate increase results in a pass for all indoor air temperature values.

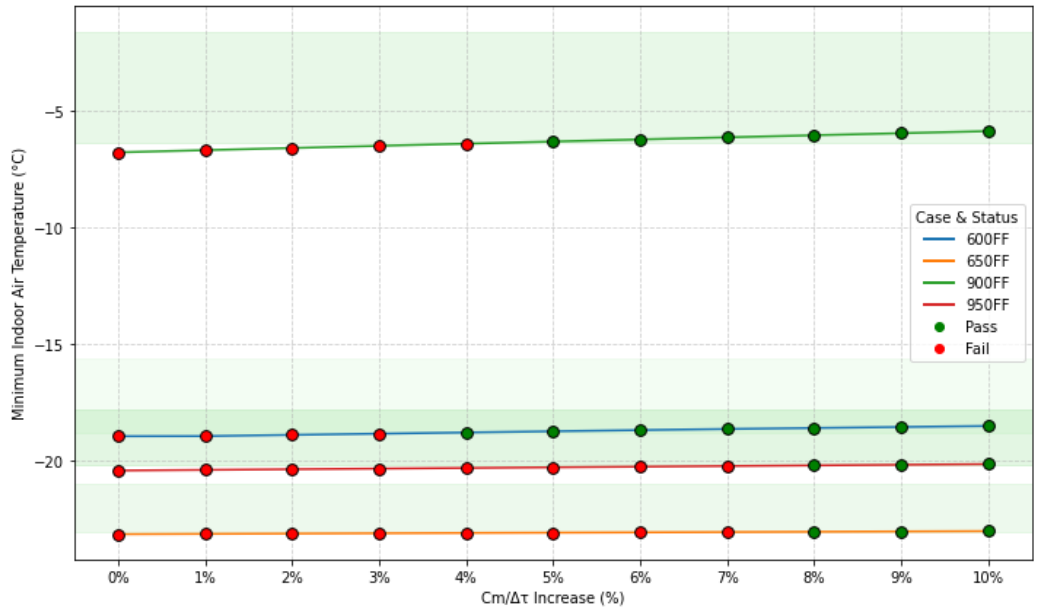

Figure 3: Sensitivity Analysis of Minimum Indoor Air Temperature to Increasing Across ASHRAE 140 Cases

Figure 3 showcases the sensitivity of the minimum indoor air temperature to gradual increases in $C_m/\Delta\tau$ for all four free-floating cases. The code to produce this plot is found in *Appendix C.2.* Notably, as the coefficient is scaled to 10% from its original value there is a linear increase in the value of air temperature. The green shading illustrates the pass/fail region for each case and the points are coloured displaying the point at which they pass. Table 5 shows that although all four cases initially failed, an 8% increase in the coefficient means they passed the benchmarks. This slight adjustment validates all cases, suggesting that the original value was an underestimation in the building's thermal capacity. This consequently improves the physical representation of the model, accounting for neglected contributions of thermal mass and dynamic effects, not included in the original grey-box model. Moreover, the cases which feature unoccupied ventilation during nighttime required a greater percentage increase to meet the temperature criterion in comparison to 600FF and 950FF. This makes sense as the lower air exchange during the night-time places greater responsibility on the internal mass to moderate the heat loss, emphasising the sensitivity of results to thermal inertia assumptions. From this analysis, the following simulations and machine learning comparisons use the first percentage increase that satisfies all cases for the ASHRAE 140 temperature criterion: 8%. This enhances the reliability and realistic capability of the model for further predictive modelling and regression evaluations.

Table 5: ASHRAE 140 Benchmarks for Each Case After 8% Increase [3]

Case	Min (°C)	Max (°C)	Avg (°C)	Status
600FF	-18.60	65.54	26.28	Pass
650FF	-22.98	63.71	19.49	Pass
900FF	-6.05	45.34	26.22	Pass
950FF	-20.20	37.87	14.53	Pass

4.1.3 Time-Series Data

To assess the different thermal behaviour of the cases, time series data was plotted for the indoor air temperature, averaging the hourly data over each day. While there are fewer data points to distinguish between day and night, the plots are more interpretable for analysis. For example, the daily average aids the visualisation of the longer-term trends by negating short-term spikes, thus the effects of mass and ventilation are also clearer. It provides plots of greater clarity and insight into seasonal dynamics. Furthermore, it showcases the effect of the ventilation schedules and thermal mass across the year.

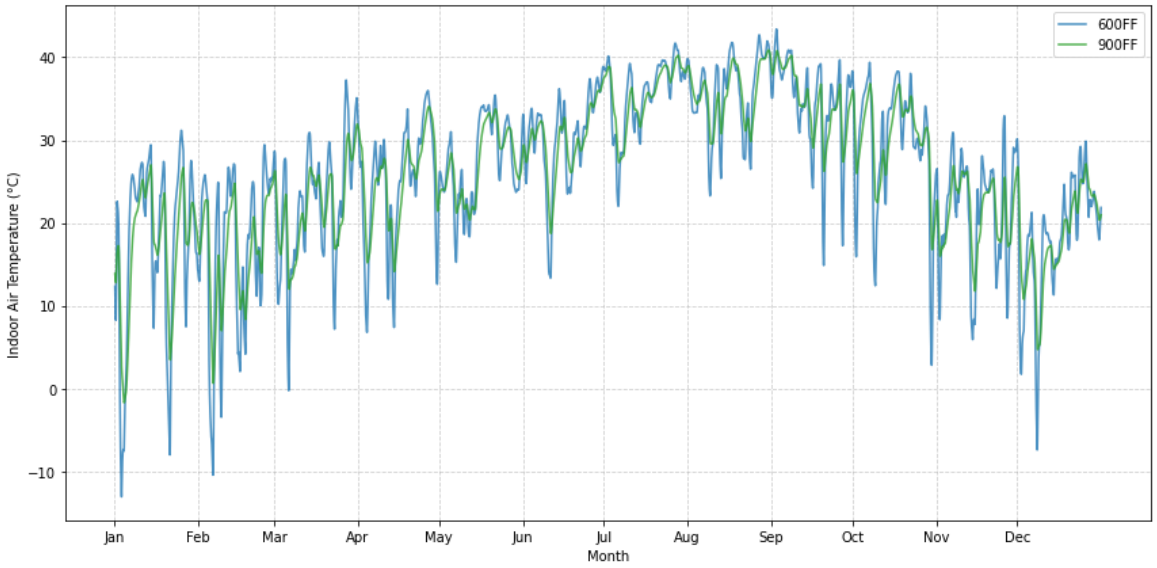

Figure 4: Daily Average Indoor Temperature Across a Calendar Year for 600FF and 900FF

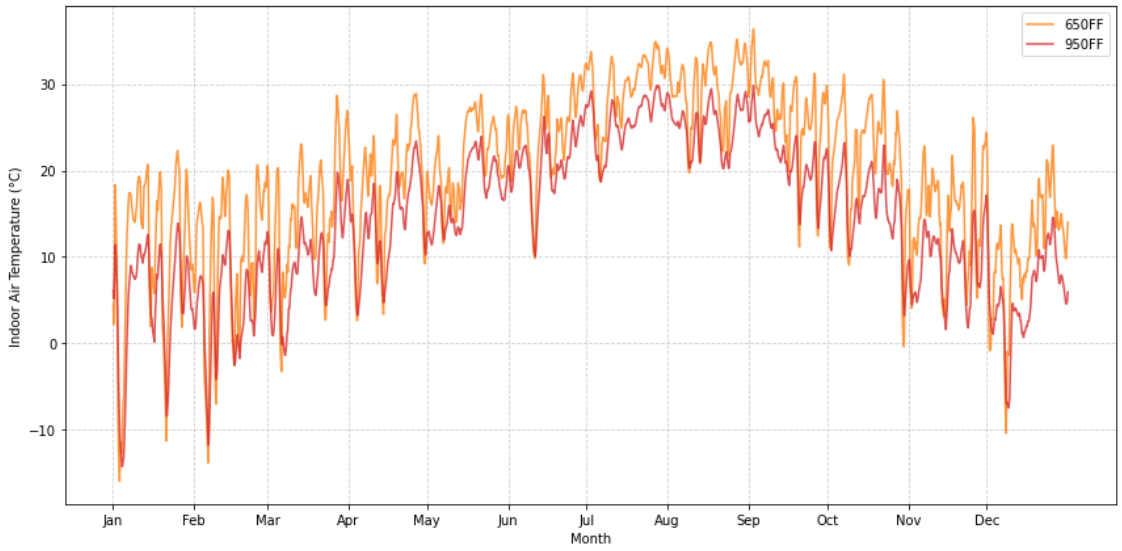

Figure 5: Daily Average Indoor Temperature Across a Calendar Year for 650FF and 950FF

Figure 4 details the time series data of indoor air temperature for cases 600FF and 900FF. These cases share the same constant occupied ventilation schedule, thereby directly comparing the effect of construction mass. Case 600FF, the lighter case, shows greater variability in the air temperature with consistently greater peaks and lower troughs. This is emphasised in the winter months where there are significantly sharper dips in comparison. Whereas, in summer with the hotter temperatures this difference is not as pronounced. Case 900FF exhibits a more stable temperature profile a phenomenon explained by the heavier construction materials providing more thermal inertia, buffering the fast outdoor temperature changes commonly seen on winter nights. This dampens the fluctuations in temperature, outlining a clear benefit of higher thermal mass in free-floating building simulations

Figure 5 compares cases 650FF and 950FF. These cases share the same dynamic ventilation schedule where there is reduced ventilation overnight. This allows the influence of the ventilation pattern to be observed while comparing the different structures. Interestingly, in comparison to constant ventilation, both cases demonstrate lower indoor air temperatures. Although reduced ventilation during the cooler nights aims to limit heat loss it is limited in the free-floating condition, where there is no heating system. In this condition, there are restricted internal gains resulting in colder temperatures in comparison. Since the unoccupied hours are from 19:00 to 07:00, this means that the lack of air exchange in the morning may limit the entry of warmer air, delaying the rebound from the overnight cooling. However, all cases display temperatures peaking during the summer and reaching the lowest during the summer, the expected seasonal trend.

Overall, the cases with the higher thermal mass exemplify a greater ability to moderate internal temperatures with reduced amplitude. The seasonal variations closely follow solar gain cycles and are consistent across all cases; this further validates the model's realism. Ultimately, although ventilation and thermal mass prove a useful tool in thermal management, their interaction must be balanced with heating control for performance optimisation. For example, a lightweight construction with an excessive ventilation schedule may inadvertently reduce thermal comfort during cold conditions. This offers valuable insight into ventilation strategies and thermal mass for future building designs. The successful performance in capturing consistent realistic data of the free-floating model provides a strong foundation for extending the methodology to scenarios with active heating and cooling.

4.2 Data-Driven Machine Learning

This section compares two data-driven regression models for indoor air temperature prediction, aligning with the main theme of interpretable and physics-informed machine learning for energy in buildings. These include tree-based models and Neural Networks (NN). For each of the ASHRAE 140 free-floating cases, the models were trained on the time series data produced for indoor air temperature against outdoor air temperature. Relating these two variables on a plot provides an important visualisation into the effect of external conditions on a building's thermal performance. This relationship is of emphasised importance in this absence of heating, where the indoor air temperature is purely influenced by the building's physics. Additionally, other factors such as the effect of mass can be analysed between the different cases. It provides a physically interpretable regression target applicable to further extensions of the model.

Tree-based models, like XGBoost [45], are a machine learning algorithm that utilise gradient boosting on decision trees, offering an interpretable accurate result. On the other hand, NN perform well at capturing non-linearities but lack explainability like tree-based models. However,

NN can be transformed into a grey-box method, pairing physical understanding and constraints of the indoor and outdoor temperature relationship. While the models are data-driven, they are trained on physically meaningful time series data, with the outputs following temperature trends in accordance with the expected thermodynamic behaviour. Regardless, there was no actual embedment of RC model constraints in the training process.

Each model used standard libraries with NN using TensorFlow's Keras API [46-48] and the tree-based model used XGboost [45]. To preprocess the data, standard scaling techniques such as Scikit's StandardScaler [49] established convergence and performance. The NN used the Adam optimiser [50] and mean squared error loss to train on the standardised input-output pairs. Its underlying architecture incorporates two hidden layers with ReLU activation functions [51]. While XGboost was trained on the same pre-processed data, it used its default tree-based splitting strategy. The code for the models producing these plots can be found in *Appendix C.3* and *Appendix C.4*.

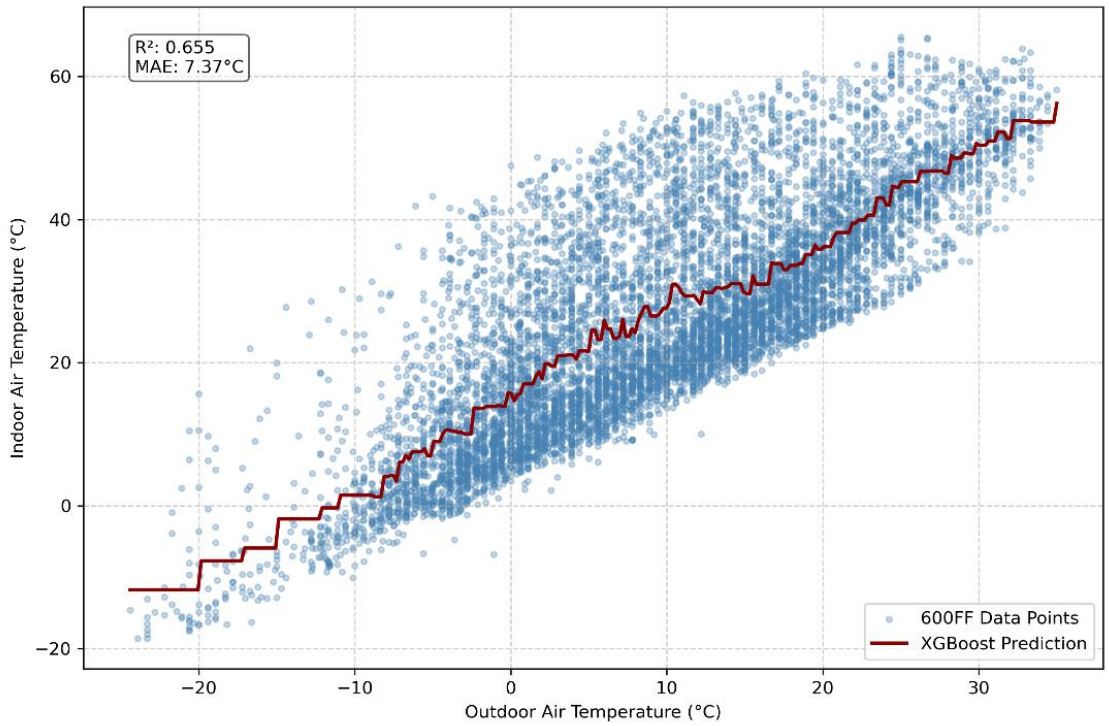

Figure 6: 600FF Tree-Based Regression Model for Indoor Air Temperature

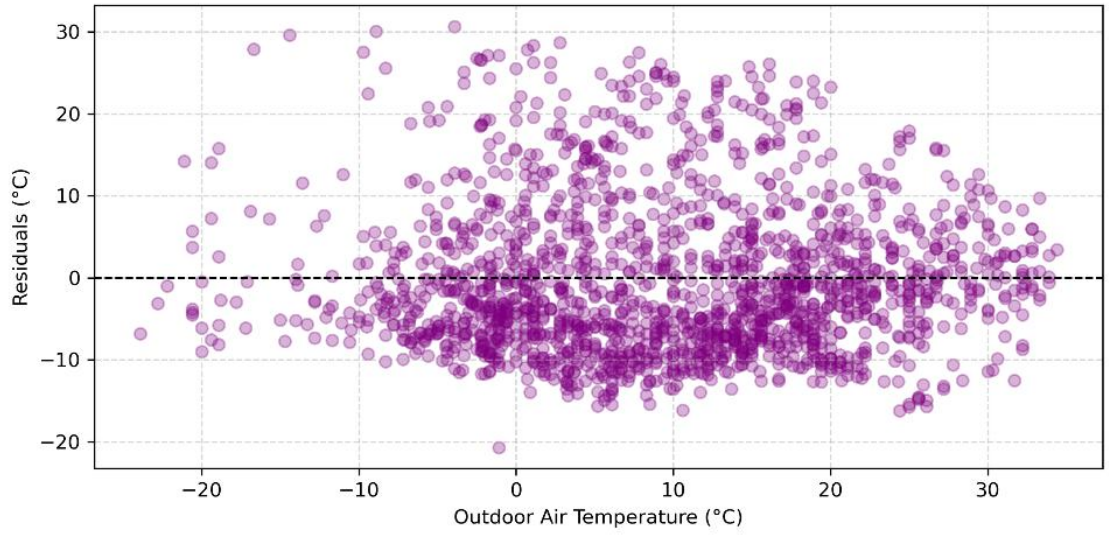

Figure 7: 600FF Tree-Based Regression Model Residuals of Indoor Air Temperature

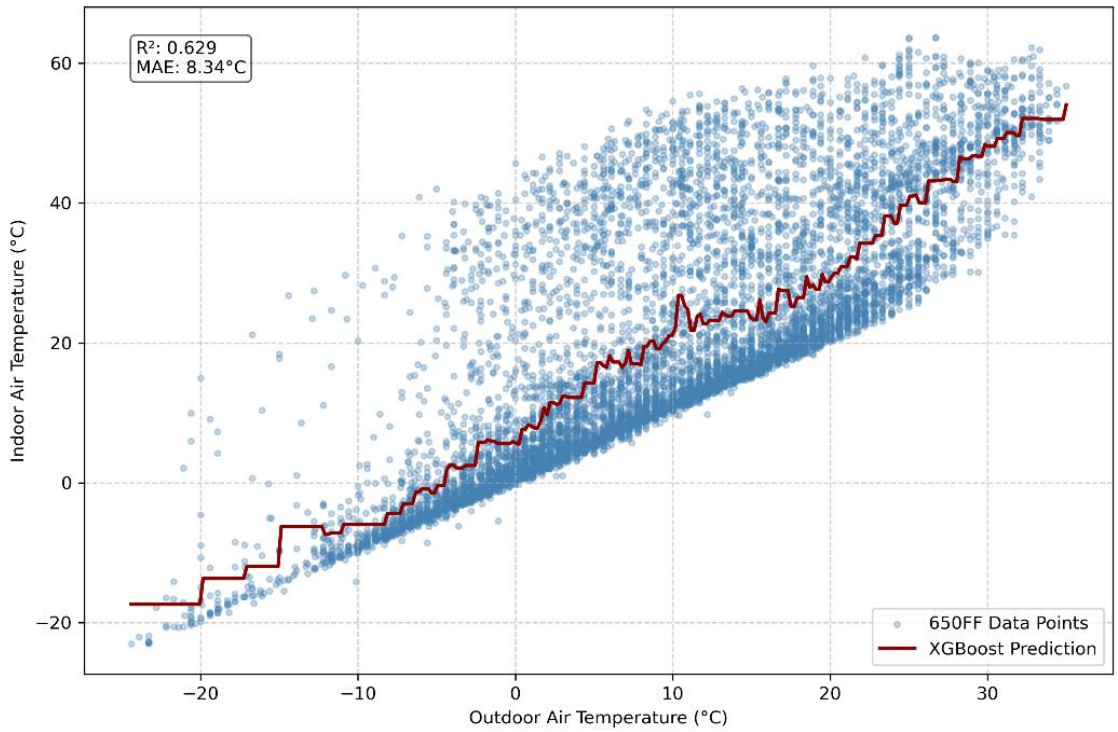

Figure 8: 650FF Tree-Based Regression Model for Indoor Air Temperature

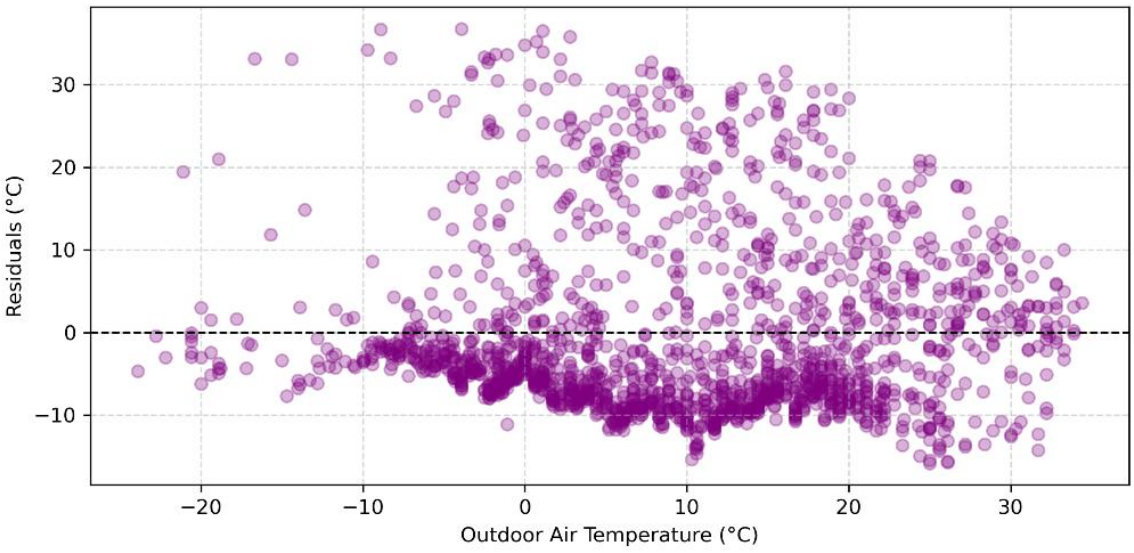

Figure 9: 650FF Tree-Based Regression Model Residuals of Indoor Air Temperature

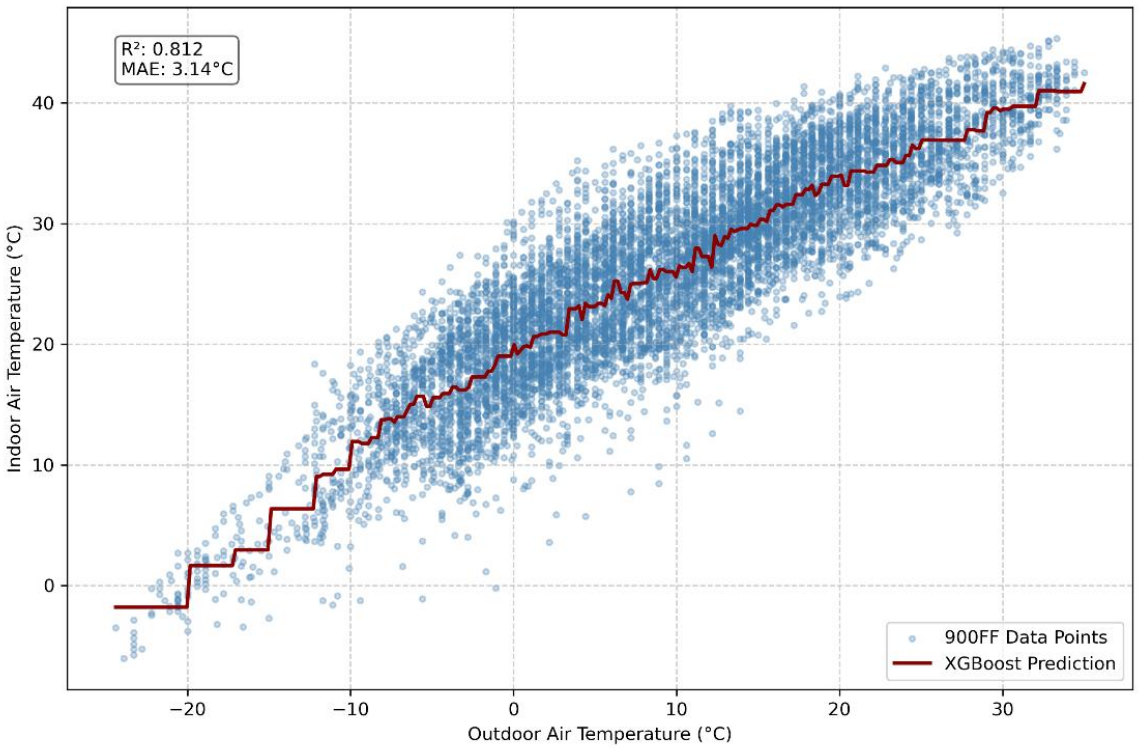

Figure 10: 900FF Tree-Based Regression Model for Indoor Air Temperature

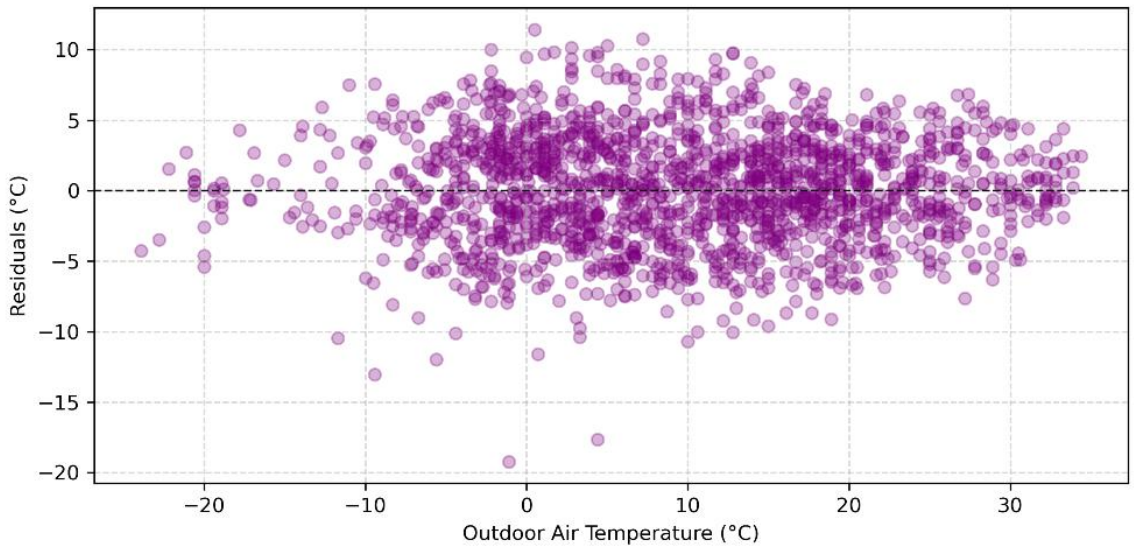

Figure 11: 900FF Tree-Based Regression Model Residuals of Indoor Air Temperature

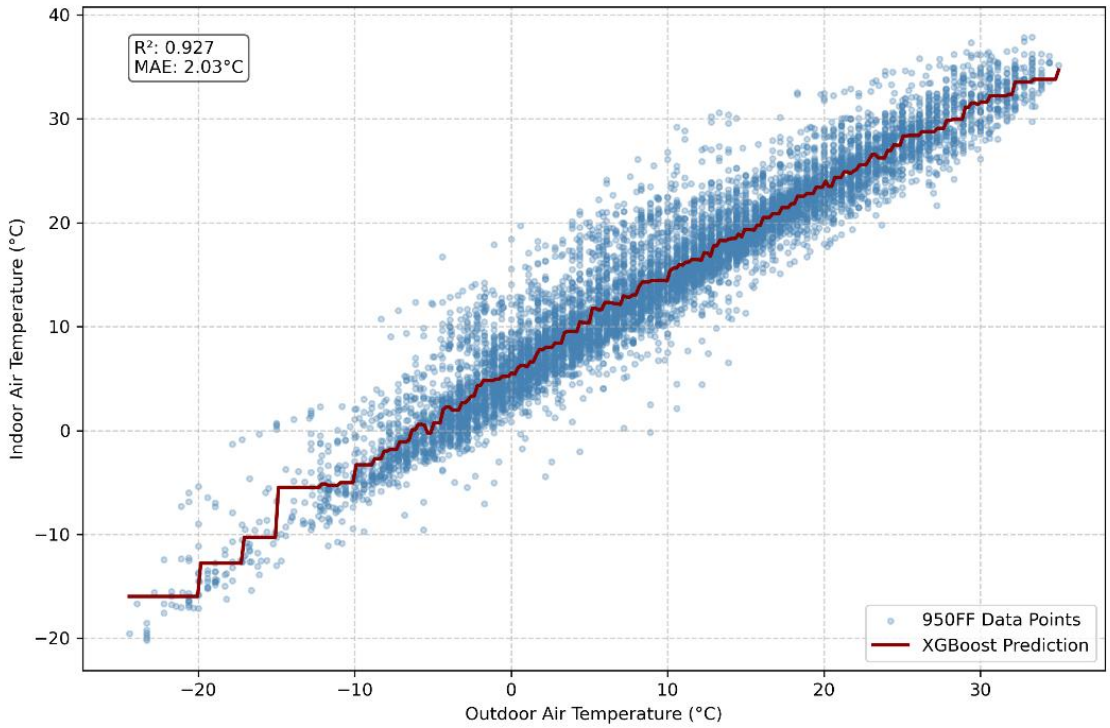

Figure 12: 950FF Tree-Based Regression Model for Indoor Air Temperature

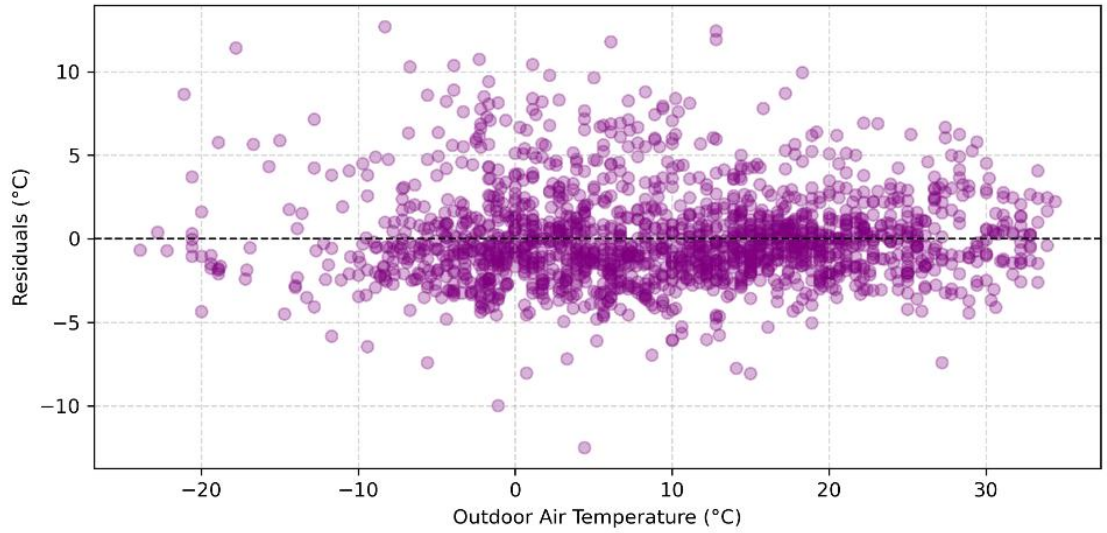

Figure 13: 950FF Tree-Based Regression Model Residuals of Indoor Air Temperature

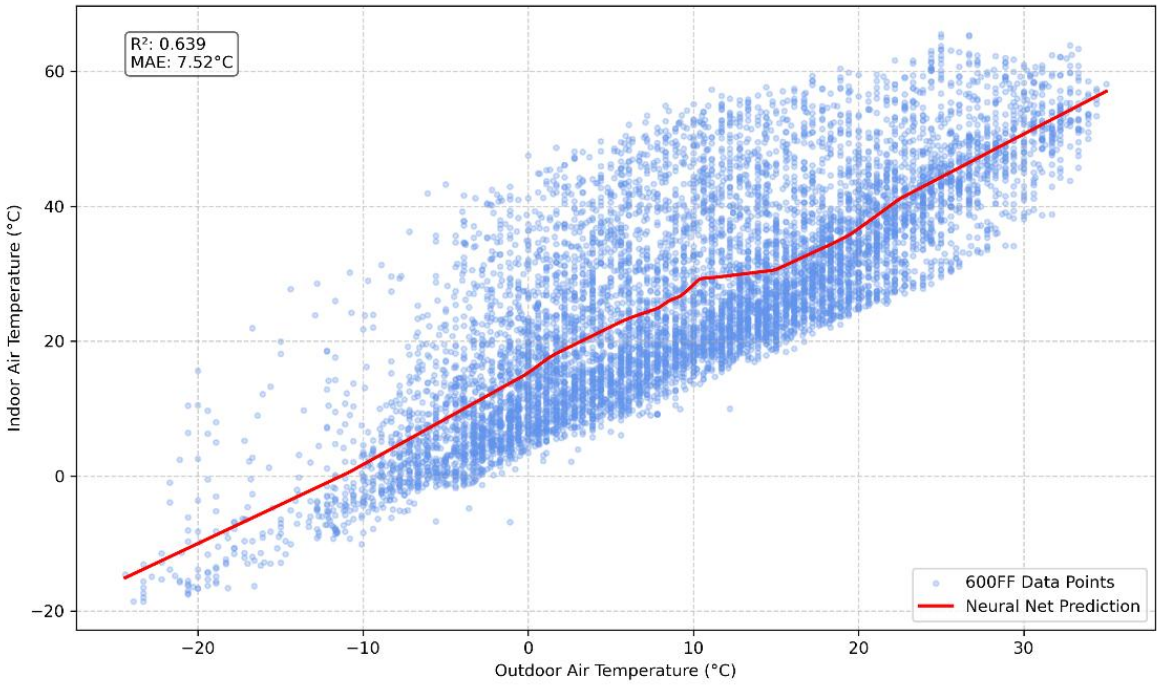

Figure 14: 600FF Neural Network Regression Model for Indoor Air Temperature

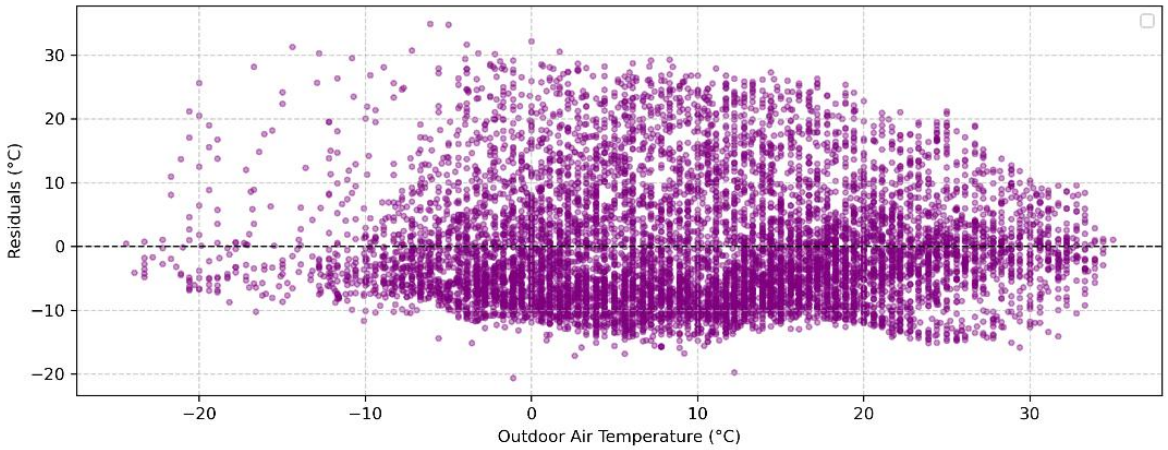

Figure 15: 600FF Neural Network Regression Model Residuals of Indoor Air Temperature

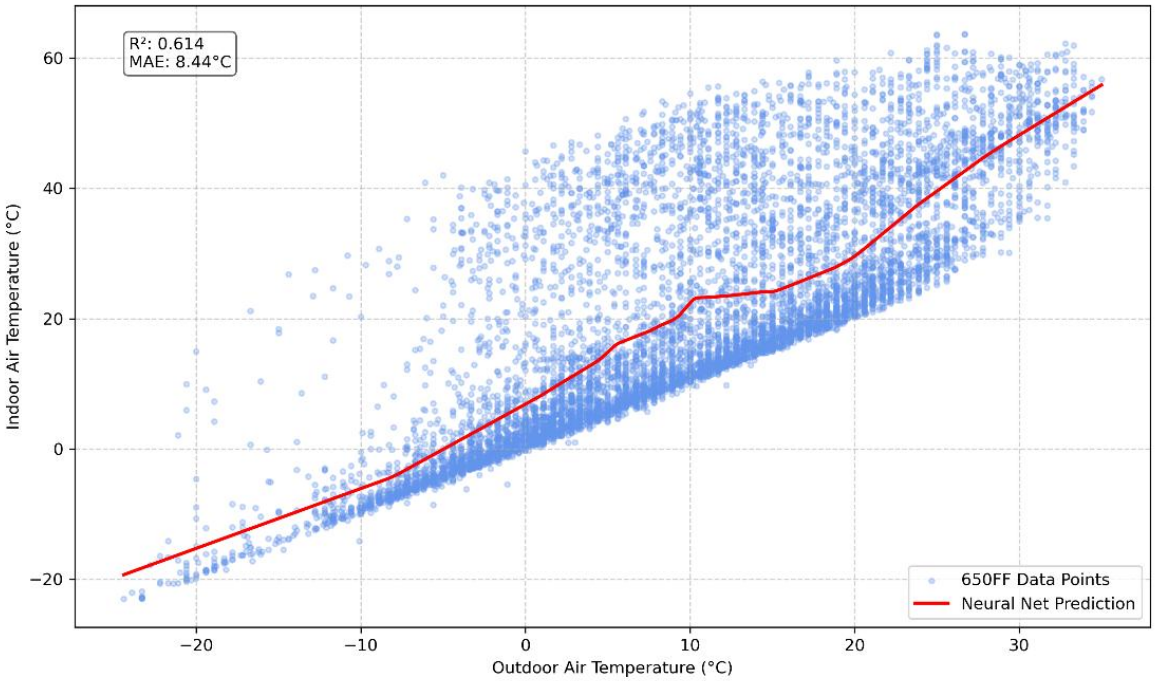

Figure 16: 650FF Neural Network Regression Model for Indoor Air Temperature

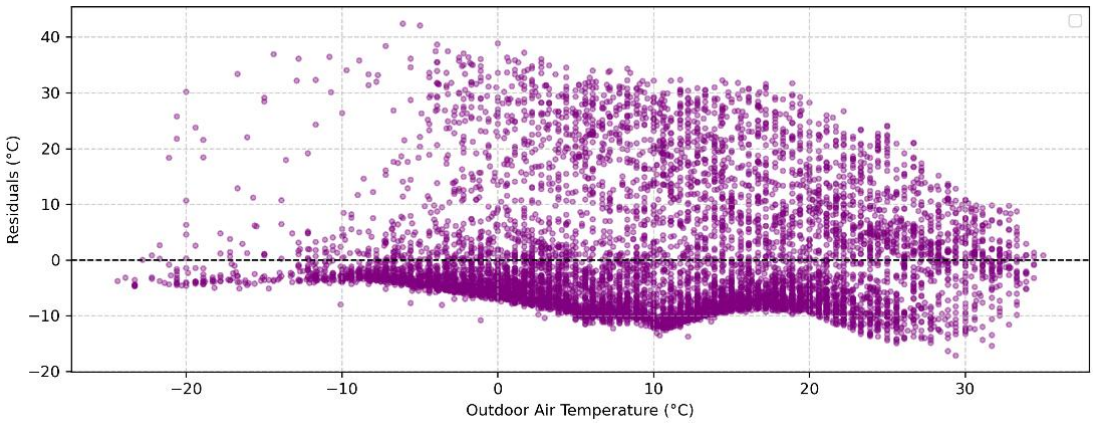

Figure 17: 650FF Neural Network Regression Model Residuals of Indoor Air Temperature

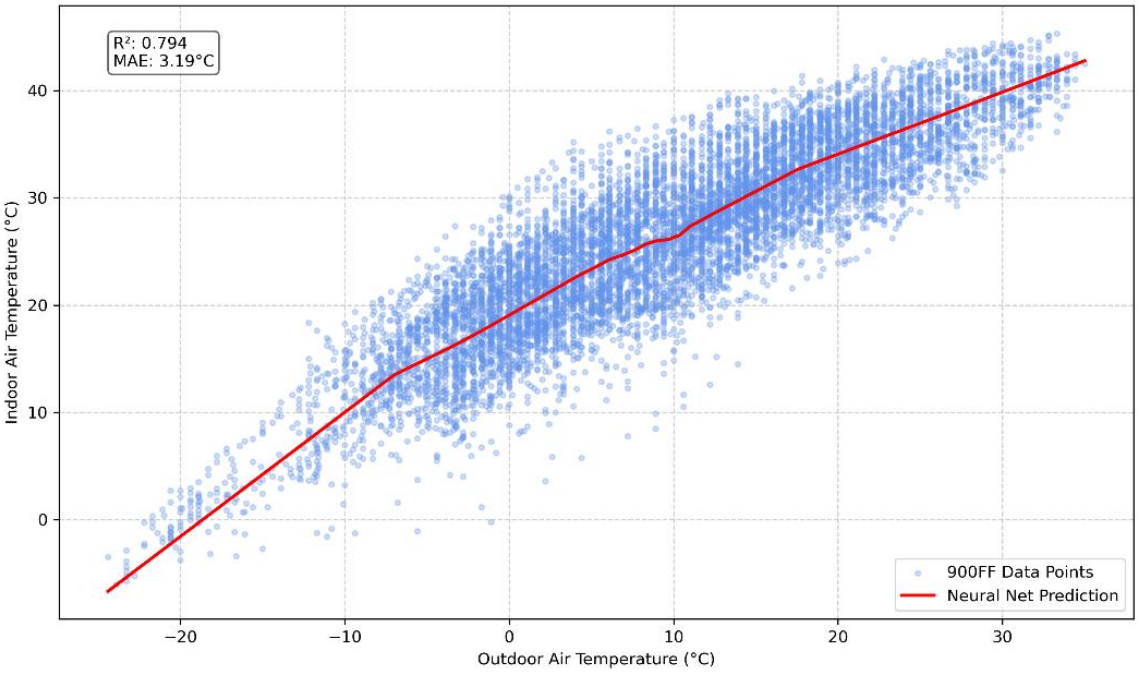

Figure 18: 900FF Neural Network Regression Model for Indoor Air Temperature

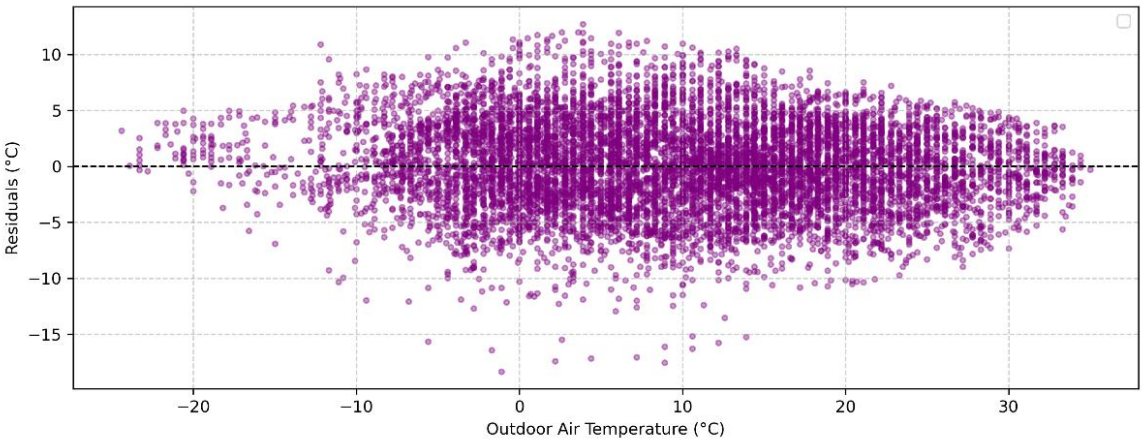

Figure 19: 900FF Neural Network Regression Model Residuals of Indoor Air Temperature

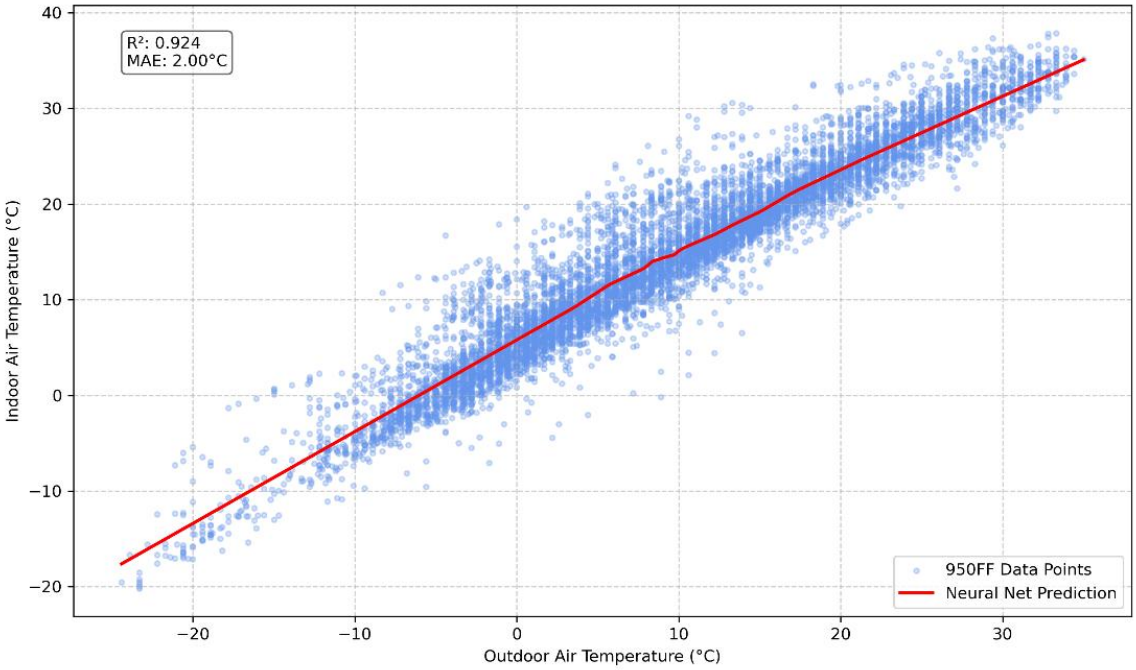

Figure 20: 950FF Neural Network Regression Model for Indoor Air Temperature

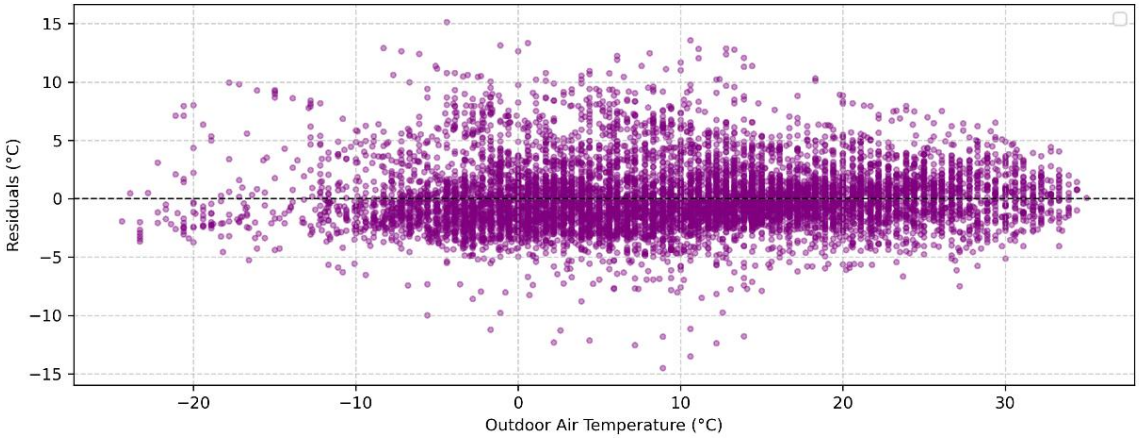

Figure 21: 950FF Neural Network Regression Model Residuals of Indoor Air Temperature

Table 6: Performative Comparison of XGBoost with NN

	Tree-Based (XGBoost)		**NN (TensorFlow Keras API)**	
Case	**R^2**	**MAE (°C)**	**R^2**	**MAE (°C)**
600FF	0.655	7.37	0.637	7.59
650FF	0.615	8.77	0.615	8.42
900FF	0.814	3.17	0.796	3.18
950FF	0.927	2.03	0.924	2.03

Figures 6-13 show the decision tree regression model and respective residuals for all cases. Figures 14-21 show the NN regression model and respective residuals for each case. Table 6 summarises the performance of the models, detailed in the legend of the plots, for comparison.

Although these models are primarily for predictive performance, they provide a visualisation of how the outdoor air temperature affects the indoor air temperature for different cases. The wider spread of data in the lower thermal mass cases (600FF and 650FF) represents the potentially lower insulation and thermal buffering. This spread of data is fed into the ML algorithms allowing informed calculations into the prediction by following the visual trend noticed. This further affirms the physical intuition into the effects of the thermal mass. The smooth prediction line of the NN is useful for continuous HVAC. On the other hand, the tree-based model, with its stepped line, compliments set-point control strategies [52]. This means that the two algorithms both have stronger applications in certain fields. Ultimately, they both possess strong predictive capabilities useful in BEM for initial design support and forecasting of temperature based on limited training data. Overall, the regression line produced is not just an illustration of the trend for interpretation but serves as a surrogate model of the building's thermal behaviour for data-informed energy management.

The plots for either regression model for all cases show the indoor air temperature against the outdoor air temperature, including the predicted curve and respective residuals on a separate plot. Table 6 shows that XGBoost performed better across all cases with a consistently lower Mean Average Error (MAE). Notably in the heavier cases, where Case 950FF had a coefficient of determination (R^2) of 0.927 and a MAE of 2.03 °C, representing a very close fit. Both models lacked accuracy in the lighter cases, where a greater spread of results was harder to model. While NN did not perform as well they produced a smoother predictive curve, useful in building energy modelling that relies on dynamic responses rather than the stepped decision boundaries from XGBoost. In terms of residual analysis, the NN produced residuals more symmetrically spread around zero. Specifically, case 950FF where greater thermal mass and ventilation schedules are present. XGBoost shows heightened levels of heteroskedasticity, specifically in the lighter cases, suggesting inaccuracy of the model outside mid-range temperatures.

Visually the vertical spread of data is much greater for the lighter mass cases as a result of the faster heat exchange. The wide range of data is more effectively captured in NN and their gradient-based learning, producing a smoother result. Both models use techniques centred around physical realism to better reflect building dynamics. XGBoost is better suited to align with thresholds and control points, useful in BEM. Whereas NN use continuous gradients that can map to the heat transfer process smoothly. Ultimately, tree-based models were selected as the regression model to use for the remainder of the study. This was driven by the motivation for interpretability and accuracy for modelling building dynamics. The interpretability is essential as it allows a user to deduce what will happen if the input changes, useful for BEM. While both models provide useful insight, XGBoost emerged as the victor due to the possibility of its traceability allowing the validation of assumptions against known physical principles [52], when applied to PIML in future work. While NN possess continuous gradient modelling power, XGBoosts superior accuracy was the ultimate deciding factor. Moreover, it fulfils the study's aim to bridge the gap between data-driven modelling and physical knowledge. This is because its tree-based structure is well suited to PIML; the decision logic can be based on embedded physical constraints. For example, if the temperature is below a certain set point value, then heating can be applied. To further evaluate the reliability of this regression model, the next section explores different uncertainty visualisation techniques for XGBoost.

4.2.1 Uncertainty Visualisation Techniques in Regression Models

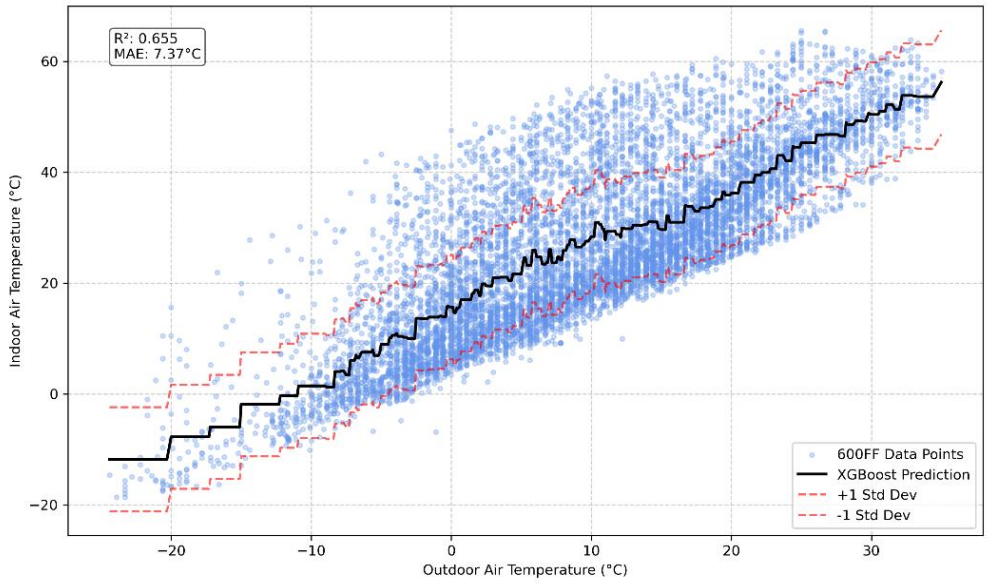

Figure 22: 600FF Standard Deviation Regression Visualisation

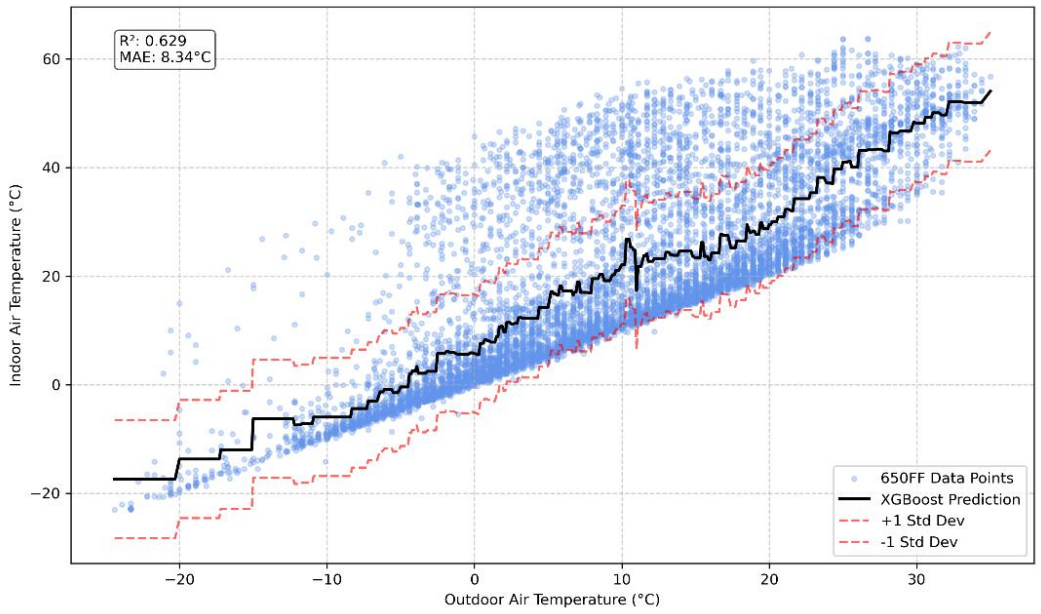

Figure 23: 650FF Standard Deviation Regression Visualisation

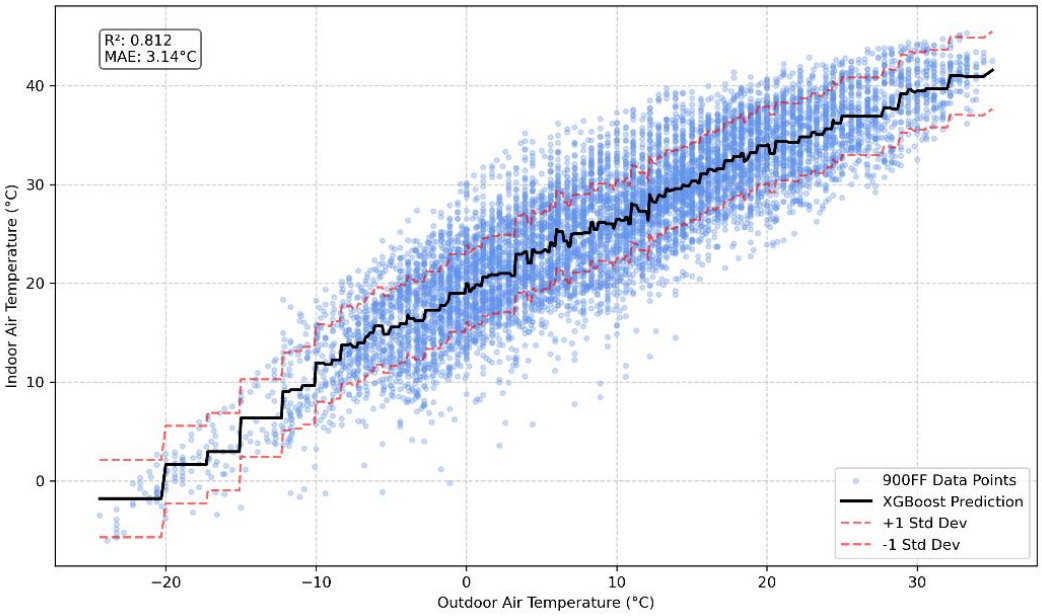

Figure 24: 900FF Standard Deviation Regression Visualisation

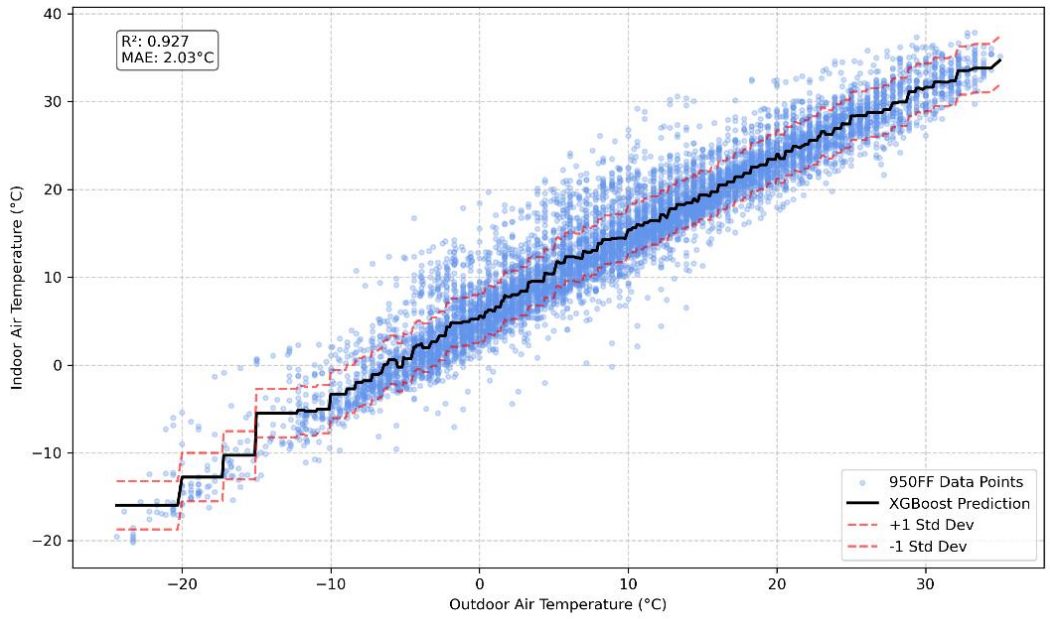

Figure 25: 950FF Standard Deviation Regression Visualisation

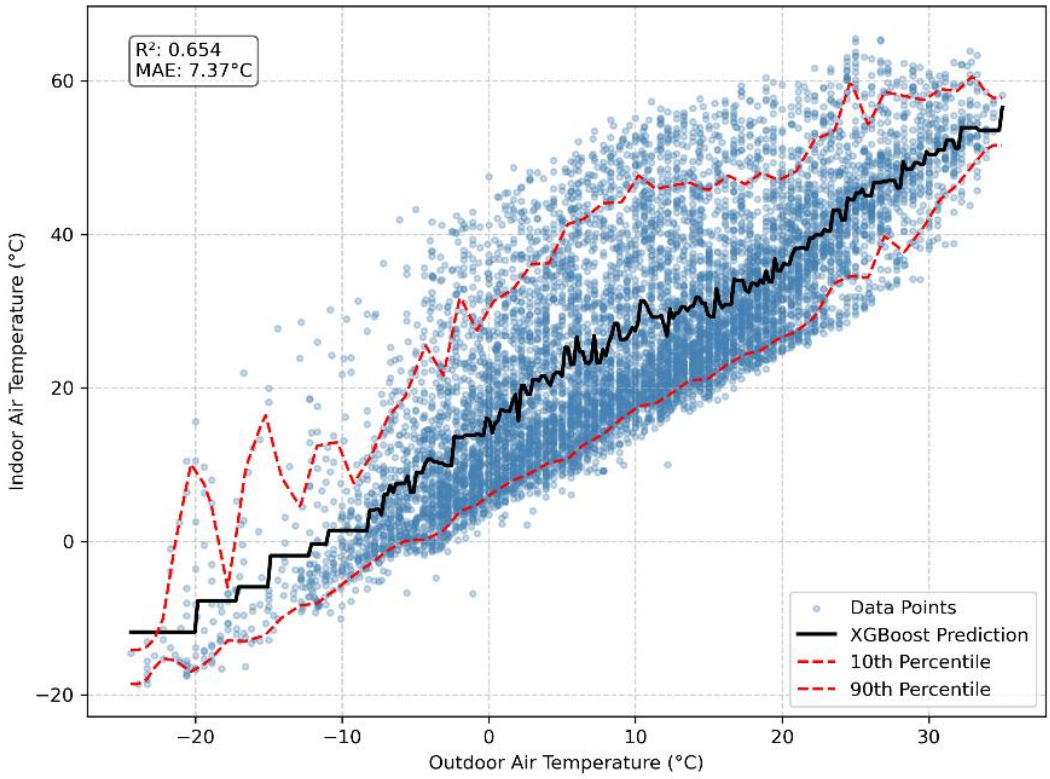

Figure 26: 600FF Quantile Band Regression Visualisation

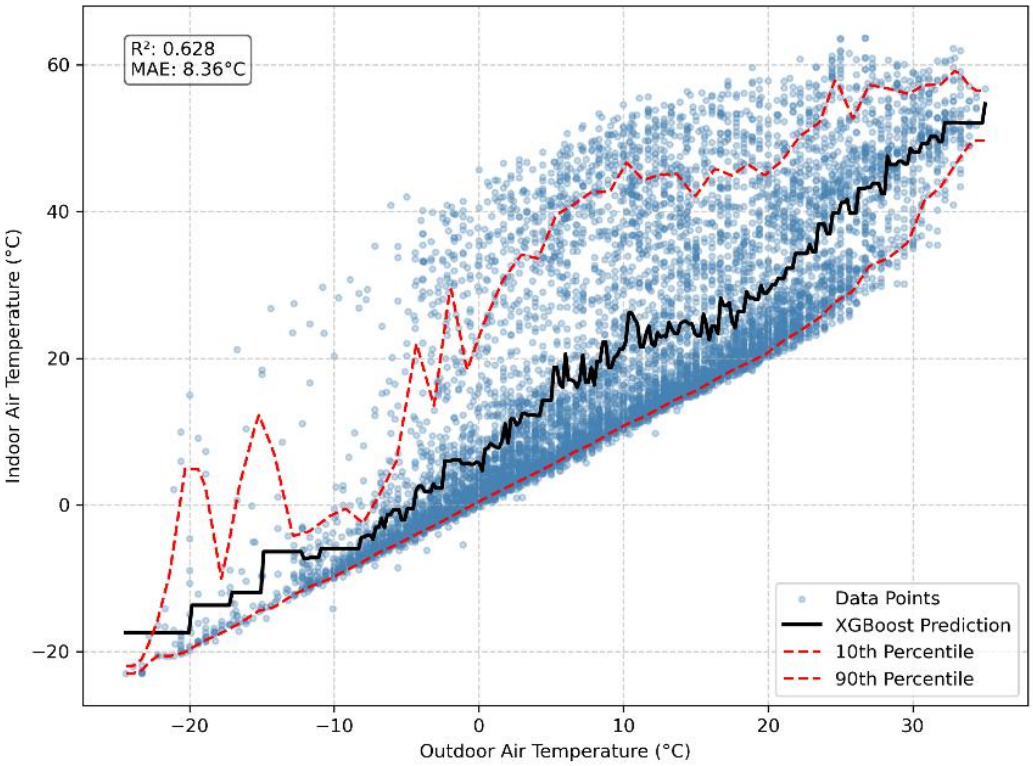

Figure 27: 650FF Quantile Band Regression Visualisation

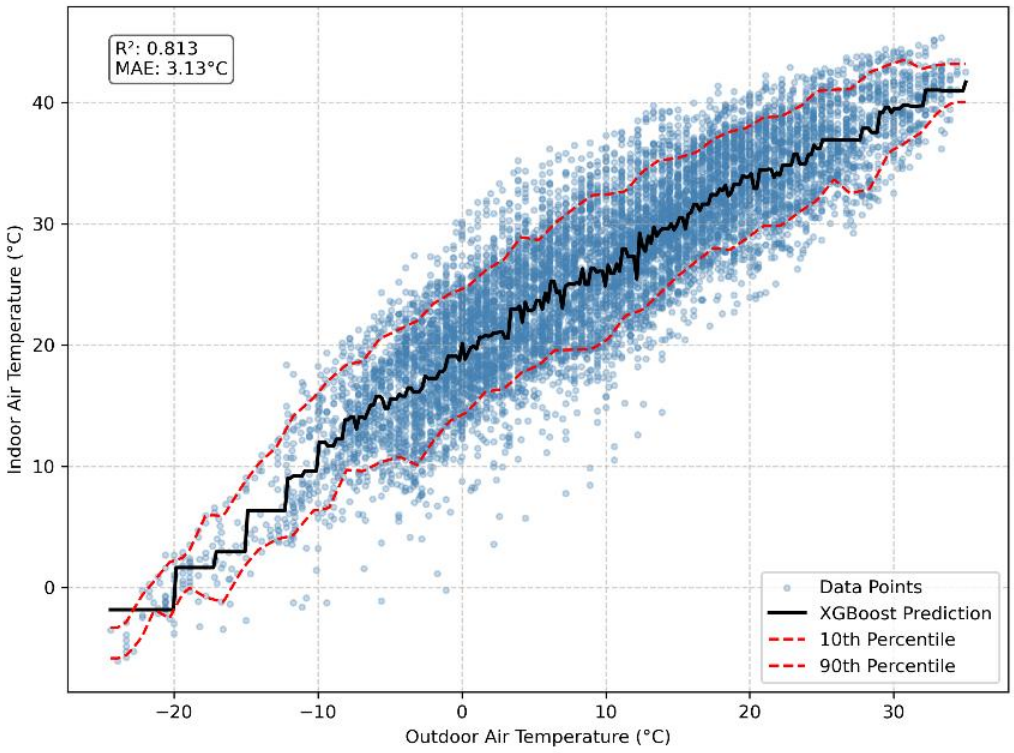

Figure 28: 900FF Quantile Band Regression Visualisation

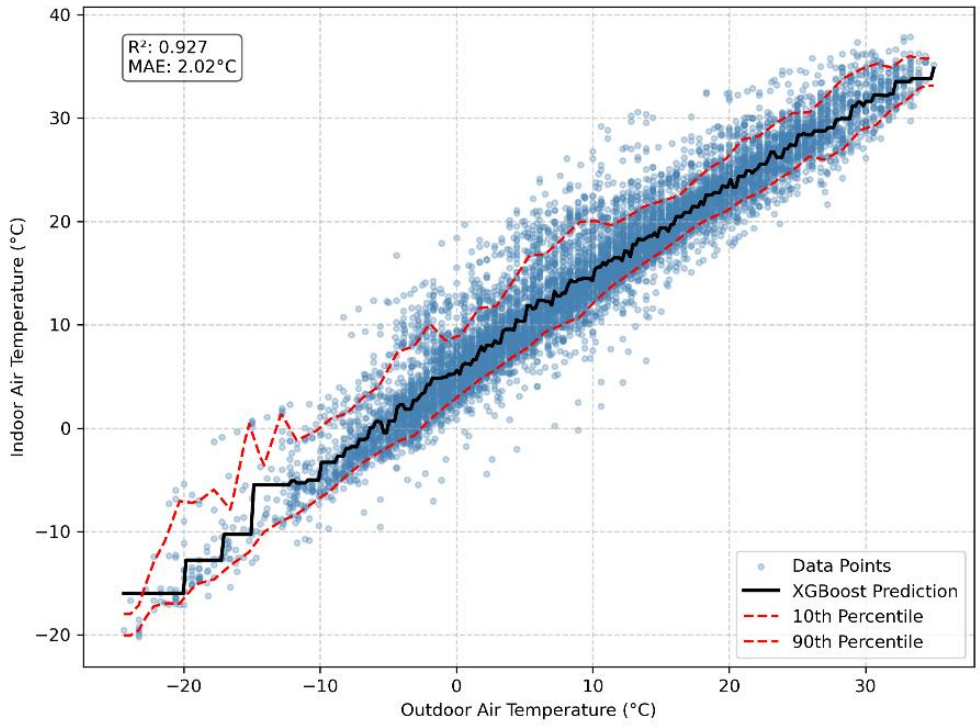

Figure 29: 950FF Quantile Band Regression Visualisation

From the selected XGBoost regression models, this section explores two uncertainty quantification methods for indoor air temperature prediction. The code to produce these plots can be found in *Appendix C.5*. These visualisations include standard deviation and percentile bands, which help deduce the model's performance, reinforcing confidence in predictions. These methods also visually display an envelope of the possible responses under varying conditions.

The standard deviation bands are based on the spread of the residuals. A rough spread around the prediction curve is produced through the calculation and plotting of standard deviation above and below the curve. This method is relatively simple and is more suitable for symmetrical residual distribution. Assuming normally distributed errors, it displays the range of the indoor temperatures at each point. This is particularly useful in BEM where thermal comfort margins can be quantified. On the other hand, percentile-based quantiles use the division of data into bins along the x-axis to compute the 10^{th} and 90^{th} percentiles of indoor temperatures. This allows for a dynamic envelope to be created that adapts to outlying data points. Additionally, this method more appropriately estimates the real-world values as it does not assume a particular error distribution. When applied, percentile bands offer a better representation of asymmetric behaviour that may be caused by varying thermal mass or ventilation.

Between the plots, it is evident that the percentile bands are a closer fit to data in the more central ranges and account for the extremes. It must be acknowledged that for the lighter cases, the 10^{th} percentile is usually jagged due to the greater spread of data and the quantile bands expand

more accurately, capturing the envelope of data better. The standard deviation bands exclude these variations and apply a constant spread. Overall, this highlights an important consideration in BEM that uncertainty visualisation should closely align with the core data characteristics. Quantile bands offer a more reliable representation of uncertainty useful in buildings characteristic with non-linear highly sensitive data. Ultimately, percentile-based uncertainty visualisation is chosen as it balances empirical insight with interpretability. It is useful for any prospective application in PIML as it ties in with the regression model and aligns with the physical variability in building thermal behaviour.

5 Conclusions and Further Work

This project has investigated a possible framework to develop Physics-Informed Machine Learning (PIML) models in the context of building energy modelling (BEM). It combines two elements, Resistance-Capacitance (RC) network models for dynamic building simulation results and Machine Learning (ML) for predictive power. One innovative element of this work is the adaptation of the standardised 5R1C model from ISO standards into grey-box modelling. The refinement of traditional assumptions produced an RC model sufficiently straightforward to solve analytically, supporting fast simulation while maintaining interpretability. This involved manipulation of the nodal energy balance, accounting better for solar and internal gains dynamics, and the incorporation of time-dependent model coefficients to avoid zero errors. This time-dependency has been handled with the use of integer variables, enabling the model to interchange between set coefficients referred to as Boolean Switching in the methodology.

The 5R1C network modelled and applied to this methodology has been validated according to ASHRAE 140 under free-floating conditions, without active heating or cooling. In initial simulations, minimum indoor air temperature narrowly failed across all cases (0.38 °C). This led to further simulation work focusing on the sensitivity of results with increases in specific coefficients. An increase of 8% to an underlying coefficient meant that values of temperature passed the benchmarks unanimously. Consequently, the following simulations were done in accordance with this. The adjustment to the coefficient was grounded in physical reasoning improving the credibility of the methodology. Validation in this free-floating state alludes to model fidelity, without HVAC the accuracy of the model is dependent on the physical realism of the RC network. Resultantly, it is safe to assume that a physical framework once validated in its base state remains valid for further implications involving heating and cooling.

The other innovation of this methodology is that it remains general and can be applied to a multitude of operational scenarios and other RC networks such as 2R1C and 7R2C. Therefore, this model reformulation becomes a scalable modelling technique for predicting indoor air temperatures, under varying thermal mass and ventilation conditions. The simulated and validated time-series data were used to train ML models. Although used in conjunction with the RC model, the testing with classical data-driven ML enabled the evaluation of the accuracy for models that do not incorporate physical constraints. The actual embedment of RC model constraints in ML creates a general PIML model.

The extension of the time-series data to ML regression models allowed the evaluation of certain techniques for interpretability and prediction accuracy. Tree-based models such as XGBoost prevailed as the more consistent performer when compared to Neural Networks (NN), mainly in the heavier thermal mass cases. Whilst NN, with their gradient smoothing, offered a better

visualisation of predictions, their sensitivity to input scaling hinders their application to strongly regulated energy scenarios. On the other hand, tree-based models are more transparent in decision structures, have greater ease of validation and can be applied to physical boundaries leading to PIML. The inclusion of quantile-based and standard deviation-based uncertainty visualisation further validates the model's predictions.

Practical applications of this framework include initial building design, retrofitting and control strategy development. As the model is grounded in physical knowledge, with fast computation and ML algorithms implemented, it aids real-time scenario evaluation across design alternatives. This means that new construction typologies, different climates or varying ventilation schedules can be incorporated with little to no change to the solver. Future work includes extended testing in these different cases and scenarios of application. Firstly, added active heating or cooling assesses the model's predictive performance under active energy management conditions, including power limits. Secondly, to improve the real-world application of the model, weather files and stochastic inputs can be applied to the simulation when there is occupant ventilation [34,52].

Concerning the evolution of data-driven ML to be physics informed, real sensor data can be used to update relevant coefficients or provide a feedback loop for correction. Identifying the best ML model formulation that incorporates the physical constraints of the RC network is essential for maintaining interpretability. While current work in this field is based mostly on deep NN [54-57], it is not necessarily limited to it. Tree-based modelling approaches, as explored in this study, are more interpretable as they can be used to identify recurrent patterns expressed as rules in the tree structure.

To conclude, this project has developed a modelling workflow combining physics-based building energy simulation and machine learning techniques. This means that the workflow is grounded in the fundamentals of both thermodynamics and machine learning. This way it can evolve further into a complete PIML modelling framework, leveraging RC formulation. Classical ML approaches guided with domain knowledge can become more accurate, insightful, and reliable in practice. The ideal PIML formulation needs to be as fast and scalable as the RC model and possess greater accuracy in approximating dynamic building operation while retaining the interpretability, intrinsic to physics-based formulation.

6 Appendix A – Additional Equations

A.1

The linear equations that form the state space matrix equation in Eq.(25) are [2]:

Equation 1 (Energy balance for the thermal mass node):

$$1+\left(\frac{H_{tr,em}+H_{tr,ms}}{\frac{C_m}{\Delta\tau}}\right)\Theta_{m,i}-\frac{H_{tr,ms}}{\frac{C_m}{\Delta\tau}}\Theta_{sup}=\Theta_{m,i-1}+\frac{H_{tr,ms}}{\frac{C_m}{\Delta\tau}}\Theta_e+\phi_{m,i}+f_m\phi_{HC.nd} \tag{29}$$

Equation 2 (Energy balance for the supply of air / surface node):

$$-H_{tr,ms}\Theta_{m,i}+(H_{tr,ms}+H_{tr,w}+H_{tr,is})\Theta_{sup}-H_{tr,is}\Theta_{air}=H_{tr,w}\Theta_e+\phi_{s,i}+f_s\phi_{HC,nd} \tag{30}$$

Equation 3 (Energy balance for the air node):

$$-H_{tr,is}\Theta_{sup}+H_{tr,is}+H_{ve}\Theta_{air}=H_{ve}\Theta_e+\phi_a+f_a\phi_{HC,nd} \tag{31}$$

A.2

Equations to solve the dynamic problem as in ISO 13790 5R1C original approach (Crank-Nicolson in the original formulation, Implicit, Explicit)[2]:

$$\theta_{m,t}=\frac{\theta_{m,t-1}\times\left[\frac{C_m}{\Delta\tau}-0{,}5\times\left(H_{tr,3}+H_{tr,em}\right)\right]+\Phi_{mtot}}{\left[\frac{C_m}{\Delta\tau}+0{,}5\times\left(H_{tr,3}+H_{tr,em}\right)\right]} \tag{32}$$

$$\Phi_{m,tot}=\Phi_m+H_{tr,em}\theta_e+\frac{H_{tr,3}\left[\Phi_s+H_{tr,es}\theta_e+H_{tr,1}\left(\theta_e+\frac{\Phi_i+f_{iH}\Phi_H-f_{iC}\Phi_C}{H_{ve}}\right)\right]}{H_{tr,2}} \tag{33}$$

$$\theta_s=\frac{H_{tr,ms}\,\theta_m+H_{tr,es}\theta_e+\Phi_s+H_{tr,1}\left[\theta_e+\frac{\Phi_i+f_{iH}\Phi_H-f_{iC}\Phi_C}{H_{ve}}\right]}{\left[H_{tr,ms}+H_{tr,es}+H_{tr,1}\right]} \tag{34}$$

$$\theta_i=\frac{H_{tr,is}\,\theta_s+H_{ve}\theta_e+\Phi_i+f_{iH}\Phi_H-f_{iC}\Phi_C}{\left[H_{tr,is}+H_{ve}\right]} \tag{35}$$

$$\theta_{op}=\theta_i f_{op}+\theta_s\left(1-f_{op}\right) \tag{36}$$

$$\theta_{op}=\frac{\theta_i\left(1-\frac{h_{ci}}{h_{rs}}\right)+\theta_i\left(1+\frac{h_{ci}}{h_{rs}}\right)}{2} \tag{37}$$

$$h_{rs}=1.2h_{ri} \tag{38}$$

Reformulation with implicit solution scheme instead of Crank Nicholson:

$$\theta_{m,t} = \frac{\theta_{m,t-1}\left[\frac{C_m}{\Delta\tau} - 0{,}5\left(H_{tr,3} + H_{tr,em}\right)\right] + \varphi_{mtot}}{\left[\frac{C_m}{\Delta\tau} + 0{,}5\left(H_{tr,3} + H_{tr,em}\right)\right]} \tag{39}$$

$$\theta_{m,i} = \frac{\frac{C_m}{\Delta\tau}\theta_{m,i-1} + H_{tr,em}\theta_e + \frac{H_{tr,3}}{H_{tr,2}}\left(H_{tr,w}\theta_e + H_{tr,w}\theta_e + \frac{H_{tr,1}}{H_{ve}}\left(\Phi_i + f_{iH}\Phi_H - f_{iC}\Phi_C\right) + \Phi_s + f_{sH}\Phi_H - f_{sC}\Phi_C\right) + \Phi_m f_{mH}\Phi_H - f_{mC}\Phi_C}{\left(H_{tr,ms} + H_{tr,es} + H_{tr,1}\right)} \tag{40}$$

$$\theta_s = \frac{H_{tr,ms}\,\theta_m + H_{tr,es}\theta_e + \Phi_s + f_{sH}\Phi_H - f_{sC}\Phi_C + H_{tr,1}\left[\theta_e + \frac{\Phi_i + f_{iH}\Phi_H - f_{iC}\Phi_C}{H_{ve}}\right]}{\left[H_{tr,ms} + H_{tr,es} + H_{tr,1}\right]} \tag{41}$$

$$\theta_i = \frac{H_{tr,is}\,\theta_s + H_{ve}\theta_e + \Phi_i + f_{iH}\Phi_H - f_{iC}\Phi_C}{\left[H_{tr,is} + H_{ve}\right]} \tag{42}$$

$$\theta_{op} = \theta_i f_{op} + \theta_s\left(1 - f_{op}\right) \tag{43}$$

$$\theta_{op} = \frac{\theta_i\left(1 - \frac{h_{ci}}{h_{rs}}\right) + \theta_i\left(1 + \frac{h_{ci}}{h_{rs}}\right)}{2} \tag{44}$$

$$h_{rs} = 1.2h_{ri} \tag{45}$$

7 Appendix B – Spreadsheets

This section details a part of the time-series data for the assembled RC thermal network grey-box model with reformulations. It is across a period of two days at the start of the calendar year. It showcases some relevant coefficients (i.e. the ventilation schedules and the heat transfer coefficients) for the reformulations in the methodology and the values of temperature for analysis, including indoor air temperature. These cases have been adjusted from the sensitivity analysis. For Case 600FF and its hourly time-steps:

Hours (incremental)	**Daily schedule ventilation**	t_e	H_{ve}	$H_{tr,1}$	$H_{tr,2}$	$H_{tr,3}$	$t_{m,i-1,model}$	$t_{m,i,model}$	$t_{s,model}$	$t_{air,model}$
[h]		[°C]	W/K	W/K	W/K	W/K		[°C]	[°C]	[°C]
1	1	0.0	17.9	17.4	53.4	50.9	15.0	12.9	12.5	12.3
2	1	0.0	17.9	17.4	53.4	50.9	12.9	11.2	10.8	10.7
3	1	0.0	17.9	17.4	53.4	50.9	11.2	9.7	9.4	9.3
4	1	0.0	17.9	17.4	53.4	50.9	9.7	8.4	8.2	8.2
5	1	0.0	17.9	17.4	53.4	50.9	8.4	7.4	7.2	7.2
6	1	0.0	17.9	17.4	53.4	50.9	7.4	6.5	6.3	6.3
7	1	3.3	17.9	17.4	53.4	50.9	6.5	6.2	6.2	6.3
8	1	2.2	17.9	17.4	53.4	50.9	6.2	5.9	5.8	5.9
9	1	3.3	17.9	17.4	53.4	50.9	5.9	7.8	7.9	8.0

10	1	7.2	17.9	17.4	53.4	50.9	7.8	9.3	9.5	9.6
11	1	7.8	17.9	17.4	53.4	50.9	9.3	19.4	19.8	19.6
12	1	8.9	17.9	17.4	53.4	50.9	19.4	30.3	30.5	30.1
13	1	9.4	17.9	17.4	53.4	50.9	30.3	39.4	39.2	38.5
14	1	10.6	17.9	17.4	53.4	50.9	39.4	46.3	45.8	44.9
15	1	10.0	17.9	17.4	53.4	50.9	46.3	49.7	48.8	47.9
16	1	9.4	17.9	17.4	53.4	50.9	49.7	49.8	48.7	47.7
17	1	6.7	17.9	17.4	53.4	50.9	49.8	44.4	43.0	42.2
18	1	5.6	17.9	17.4	53.4	50.9	44.4	38.8	37.5	36.8
19	1	2.8	17.9	17.4	53.4	50.9	38.8	33.6	32.4	31.8
20	1	2.2	17.9	17.4	53.4	50.9	33.6	29.1	28.1	27.5
21	1	1.7	17.9	17.4	53.4	50.9	29.1	25.2	24.3	23.9
22	1	0.6	17.9	17.4	53.4	50.9	25.2	21.7	20.9	20.5
23	1	0.0	17.9	17.4	53.4	50.9	21.7	18.7	18.0	17.7
24	1	-4.4	17.9	17.4	53.4	50.9	18.7	15.4	14.7	14.3
25	1	-5.6	17.9	17.4	53.4	50.9	15.4	12.5	11.8	11.5
26	1	-7.2	17.9	17.4	53.4	50.9	12.5	9.7	9.1	8.8
27	1	-8.9	17.9	17.4	53.4	50.9	9.7	7.1	6.5	6.3
28	1	-9.4	17.9	17.4	53.4	50.9	7.1	4.8	4.3	4.1
29	1	-10.6	17.9	17.4	53.4	50.9	4.8	2.7	2.3	2.1
30	1	-11.7	17.9	17.4	53.4	50.9	2.7	0.7	0.3	0.2
31	1	-12.2	17.9	17.4	53.4	50.9	0.7	-1.0	-1.4	-1.5
32	1	-12.2	17.9	17.4	53.4	50.9	-1.0	-2.5	-2.8	-2.9
33	1	-12.8	17.9	17.4	53.4	50.9	-2.5	-3.6	-3.9	-3.9
34	1	-12.8	17.9	17.4	53.4	50.9	-3.6	-4.4	-4.6	-4.6
35	1	-12.8	17.9	17.4	53.4	50.9	-4.4	-4.9	-5.1	-5.1
36	1	-13.9	17.9	17.4	53.4	50.9	-4.9	-5.5	-5.7	-5.7
37	1	-15.0	17.9	17.4	53.4	50.9	-5.5	-6.1	-6.3	-6.4
38	1	-15.0	17.9	17.4	53.4	50.9	-6.1	-6.7	-6.9	-7.0
39	1	-15.0	17.9	17.4	53.4	50.9	-6.7	-7.4	-7.5	-7.6
40	1	-16.1	17.9	17.4	53.4	50.9	-7.4	-8.2	-8.5	-8.5
41	1	-17.2	17.9	17.4	53.4	50.9	-8.2	-9.4	-9.6	-9.6
42	1	-17.8	17.9	17.4	53.4	50.9	-9.4	-10.4	-10.6	-10.7
43	1	-17.8	17.9	17.4	53.4	50.9	-10.4	-11.3	-11.5	-11.5
44	1	-18.3	17.9	17.4	53.4	50.9	-11.3	-12.2	-12.4	-12.3
45	1	-18.3	17.9	17.4	53.4	50.9	-12.2	-12.9	-13.1	-13.0
46	1	-18.9	17.9	17.4	53.4	50.9	-12.9	-13.6	-13.8	-13.7
47	1	-19.4	17.9	17.4	53.4	50.9	-13.6	-14.3	-14.4	-14.4
48	1	-18.9	17.9	17.4	53.4	50.9	-14.3	-14.8	-14.9	-14.8

For Case 650FF, note the varying ventilation schedule:

Hours (incremental)	**Daily schedule ventilation**	**H_{ve}**	**$H_{tr,1}$**	**t_e**	**$H_{tr,2}$**	**$H_{tr,3}$**	**$t_{m,i-1,model}$**	**$t_{m,i,model}$**	**$t_{s,model}$**	**$t_{air,model}$**	**$t_{op,model}$**
[h]		W/K	W/K	[°C]	W/K	W/K		[°C]	[°C]	[°C]	[°C]
1	0	488.5	267.7	0.0	303.7	243.8	15.0	10.0	8.1	4.6	0.0
2	0	488.5	267.7	0.0	303.7	243.8	10.0	6.8	5.5	3.1	0.0
3	0	488.5	267.7	0.0	303.7	243.8	6.8	4.6	3.7	2.1	0.0
4	0	488.5	267.7	0.0	303.7	243.8	4.6	3.1	2.5	1.5	0.0
5	0	488.5	267.7	0.0	303.7	243.8	3.1	2.1	1.8	1.1	0.0
6	0	488.5	267.7	0.0	303.7	243.8	2.1	1.5	1.3	0.8	0.0
7	0	488.5	267.7	3.3	303.7	243.8	1.5	2.2	2.5	2.9	0.0
8	1	17.9	17.4	2.2	53.4	51.2	2.2	2.5	2.6	2.8	0.0
9	1	17.9	17.4	3.3	53.4	51.2	2.5	5.0	5.2	5.3	0.0
10	1	17.9	17.4	7.2	53.4	51.2	5.0	6.9	7.1	7.3	0.0
11	1	17.9	17.4	7.8	53.4	51.2	6.9	17.3	17.8	17.7	0.0
12	1	17.9	17.4	8.9	53.4	51.2	17.3	28.6	28.8	28.4	0.0
13	1	17.9	17.4	9.4	53.4	51.2	28.6	37.9	37.7	37.1	0.0
14	1	17.9	17.4	10.6	53.4	51.2	37.9	45.0	44.5	43.7	0.0
15	1	17.9	17.4	10.0	53.4	51.2	45.0	48.6	47.8	46.9	0.0
16	1	17.9	17.4	9.4	53.4	51.2	48.6	48.8	47.8	46.8	0.0
17	1	17.9	17.4	6.7	53.4	51.2	48.8	43.7	42.3	41.4	0.0
18	1	17.9	17.4	5.6	53.4	51.2	43.7	38.2	36.9	36.2	0.0
19	0	488.5	267.7	2.8	303.7	243.8	38.2	26.4	21.8	13.3	0.0
20	0	488.5	267.7	2.2	303.7	243.8	26.4	18.3	15.2	9.4	0.0
21	0	488.5	267.7	1.7	303.7	243.8	18.3	12.8	10.7	6.7	0.0
22	0	488.5	267.7	0.6	303.7	243.8	12.8	8.8	7.2	4.3	0.0
23	0	488.5	267.7	0.0	303.7	243.8	8.8	5.9	4.8	2.7	0.0
24	0	488.5	267.7	-4.4	303.7	243.8	5.9	2.5	1.2	-1.2	0.0
25	0	488.5	267.7	-5.6	303.7	243.8	2.5	-0.1	-1.1	-3.0	0.0
26	0	488.5	267.7	-7.2	303.7	243.8	-0.1	-2.4	-3.3	-5.0	0.0
27	0	488.5	267.7	-8.9	303.7	243.8	-2.4	-4.5	-5.3	-6.8	0.0
28	0	488.5	267.7	-9.4	303.7	243.8	-4.5	-6.1	-6.7	-7.8	0.0
29	0	488.5	267.7	-10.6	303.7	243.8	-6.1	-7.5	-8.1	-9.1	0.0
30	0	488.5	267.7	-11.7	303.7	243.8	-7.5	-8.8	-9.4	-10.3	0.0
31	0	488.5	267.7	-12.2	303.7	243.8	-8.8	-9.9	-10.3	-11.0	0.0

32	1	17.9	17.4	-12.2	53.4	51.2	-9.9	-10.0	-10.0	-9.9	0.0
33	1	17.9	17.4	-12.8	53.4	51.2	-10.0	-10.0	-10.0	-9.9	0.0
34	1	17.9	17.4	-12.8	53.4	51.2	-10.0	-9.8	-9.8	-9.7	0.0
35	1	17.9	17.4	-12.8	53.4	51.2	-9.8	-9.6	-9.6	-9.5	0.0
36	1	17.9	17.4	-13.9	53.4	51.2	-9.6	-9.4	-9.5	-9.4	0.0
37	1	17.9	17.4	-15.0	53.4	51.2	-9.4	-9.5	-9.6	-9.5	0.0
38	1	17.9	17.4	-15.0	53.4	51.2	-9.5	-9.6	-9.7	-9.6	0.0
39	1	17.9	17.4	-15.0	53.4	51.2	-9.6	-9.8	-9.9	-9.8	0.0
40	1	17.9	17.4	-16.1	53.4	51.2	-9.8	-10.3	-10.4	-10.4	0.0
41	1	17.9	17.4	-17.2	53.4	51.2	-10.3	-11.1	-11.3	-11.3	0.0
42	1	17.9	17.4	-17.8	53.4	51.2	-11.1	-11.9	-12.1	-12.1	0.0
43	0	488.5	267.7	-17.8	303.7	243.8	-11.9	-13.8	-14.6	-15.9	0.0
44	0	488.5	267.7	-18.3	303.7	243.8	-13.8	-15.2	-15.8	-16.8	0.0
45	0	488.5	267.7	-18.3	303.7	243.8	-15.2	-16.2	-16.6	-17.2	0.0
46	0	488.5	267.7	-18.9	303.7	243.8	-16.2	-17.0	-17.3	-17.9	0.0
47	0	488.5	267.7	-19.4	303.7	243.8	-17.0	-17.7	-18.0	-18.5	0.0
48	0	488.5	267.7	-18.9	303.7	243.8	-17.7	-18.1	-18.2	-18.4	0.0

For Case 900FF:

Hours (incremental)	**Daily schedule ventilation**	**t_e**	**H_{ve}**	**$H_{tr,1}$**	**$H_{tr,2}$**	**$H_{tr,3}$**	**$t_{m,i-1,model}$**	**$t_{m,i,model}$**	**$t_{s,model}$**	**$t_{air,model}$**
[h]		[°C]	W/K	W/K	W/K	W/K		[°C]	[°C]	[°C]
1	1	0.0	17.9	17.4	53.4	50.9	15.0	14.6	14.1	13.9
2	1	0.0	17.9	17.4	53.4	50.9	14.6	14.3	13.7	13.5
3	1	0.0	17.9	17.4	53.4	50.9	14.3	13.9	13.4	13.2
4	1	0.0	17.9	17.4	53.4	50.9	13.9	13.6	13.1	12.9

5	1	0.0	17.9	17.4	53.4	50.9	13.6	13.3	12.8	12.6
6	1	0.0	17.9	17.4	53.4	50.9	13.3	12.9	12.5	12.3
7	1	3.3	17.9	17.4	53.4	50.9	12.9	12.7	12.4	12.3
8	1	2.2	17.9	17.4	53.4	50.9	12.7	12.5	12.1	12.0
9	1	3.3	17.9	17.4	53.4	50.9	12.5	12.7	12.6	12.6
10	1	7.2	17.9	17.4	53.4	50.9	12.7	12.8	12.8	12.9
11	1	7.8	17.9	17.4	53.4	50.9	12.8	14.5	15.7	15.6
12	1	8.9	17.9	17.4	53.4	50.9	14.5	16.6	18.0	17.9
13	1	9.4	17.9	17.4	53.4	50.9	16.6	18.5	19.8	19.7
14	1	10.6	17.9	17.4	53.4	50.9	18.5	20.3	21.5	21.3
15	1	10.0	17.9	17.4	53.4	50.9	20.3	21.6	22.4	22.2
16	1	9.4	17.9	17.4	53.4	50.9	21.6	22.4	22.7	22.5
17	1	6.7	17.9	17.4	53.4	50.9	22.4	22.1	21.7	21.4
18	1	5.6	17.9	17.4	53.4	50.9	22.1	21.7	21.1	20.8
19	1	2.8	17.9	17.4	53.4	50.9	21.7	21.3	20.5	20.2
20	1	2.2	17.9	17.4	53.4	50.9	21.3	20.8	20.0	19.7
21	1	1.7	17.9	17.4	53.4	50.9	20.8	20.3	19.6	19.2
22	1	0.6	17.9	17.4	53.4	50.9	20.3	19.8	19.0	18.7
23	1	0.0	17.9	17.4	53.4	50.9	19.8	19.3	18.5	18.2
24	1	-4.4	17.9	17.4	53.4	50.9	19.3	18.7	17.8	17.3
25	1	-5.6	17.9	17.4	53.4	50.9	18.7	18.1	17.1	16.7
26	1	-7.2	17.9	17.4	53.4	50.9	18.1	17.5	16.4	15.9
27	1	-8.9	17.9	17.4	53.4	50.9	17.5	16.8	15.7	15.2
28	1	-9.4	17.9	17.4	53.4	50.9	16.8	16.1	15.1	14.5
29	1	-10.6	17.9	17.4	53.4	50.9	16.1	15.4	14.4	13.8
30	1	-11.7	17.9	17.4	53.4	50.9	15.4	14.7	13.6	13.1
31	1	-12.2	17.9	17.4	53.4	50.9	14.7	14.1	13.0	12.4
32	1	-12.2	17.9	17.4	53.4	50.9	14.1	13.4	12.3	11.8
33	1	-12.8	17.9	17.4	53.4	50.9	13.4	12.8	11.7	11.2
34	1	-12.8	17.9	17.4	53.4	50.9	12.8	12.2	11.2	10.7
35	1	-12.8	17.9	17.4	53.4	50.9	12.2	11.7	10.7	10.2
36	1	-13.9	17.9	17.4	53.4	50.9	11.7	11.1	10.2	9.6
37	1	-15.0	17.9	17.4	53.4	50.9	11.1	10.6	9.6	9.0
38	1	-15.0	17.9	17.4	53.4	50.9	10.6	10.0	9.0	8.5
39	1	-15.0	17.9	17.4	53.4	50.9	10.0	9.4	8.5	8.0
40	1	-16.1	17.9	17.4	53.4	50.9	9.4	8.8	7.8	7.3
41	1	-17.2	17.9	17.4	53.4	50.9	8.8	8.2	7.1	6.6
42	1	-17.8	17.9	17.4	53.4	50.9	8.2	7.5	6.5	5.9
43	1	-17.8	17.9	17.4	53.4	50.9	7.5	6.9	5.8	5.3
44	1	-18.3	17.9	17.4	53.4	50.9	6.9	6.2	5.2	4.7
45	1	-18.3	17.9	17.4	53.4	50.9	6.2	5.6	4.6	4.1
46	1	-18.9	17.9	17.4	53.4	50.9	5.6	5.0	4.0	3.5

47	1	-19.4	17.9	17.4	53.4	50.9	5.0	4.4	3.4	2.9
48	1	-18.9	17.9	17.4	53.4	50.9	4.4	3.8	2.8	2.4

For Case 950FF, note the varying ventilation schedule:

Hours (incremental)	**Daily schedule ventilation**	t_e	H_{ve}	$H_{tr,1}$	$H_{tr,2}$	$H_{tr,3}$	$t_{m,i-1,model}$	$t_{m,i,model}$	$t_{s,model}$	$t_{air,model}$
[h]		[°C]	W/K	W/K	W/K	W/K		[°C]	[°C]	[°C]
1	0	0.0	488.5	267.7	303.7	238.6	15.0	13.9	11.0	6.1
2	0	0.0	488.5	267.7	303.7	238.6	13.9	12.9	10.2	5.7
3	0	0.0	488.5	267.7	303.7	238.6	12.9	12.0	9.5	5.3
4	0	0.0	488.5	267.7	303.7	238.6	12.0	11.2	8.8	5.0
5	0	0.0	488.5	267.7	303.7	238.6	11.2	10.4	8.2	4.6
6	0	0.0	488.5	267.7	303.7	238.6	10.4	9.7	7.6	4.3
7	0	3.3	488.5	267.7	303.7	238.6	9.7	9.2	8.0	6.0
8	1	2.2	17.9	17.4	53.4	50.9	9.2	9.1	8.9	8.9
9	1	3.3	17.9	17.4	53.4	50.9	9.1	9.4	9.5	9.5
10	1	7.2	17.9	17.4	53.4	50.9	9.4	9.6	9.8	9.9
11	1	7.8	17.9	17.4	53.4	50.9	9.6	11.4	12.7	12.7
12	1	8.9	17.9	17.4	53.4	50.9	11.4	13.5	15.1	15.1
13	1	9.4	17.9	17.4	53.4	50.9	13.5	15.6	17.0	17.0
14	1	10.6	17.9	17.4	53.4	50.9	15.6	17.4	18.7	18.7
15	1	10.0	17.9	17.4	53.4	50.9	17.4	18.8	19.7	19.6
16	1	9.4	17.9	17.4	53.4	50.9	18.8	19.6	20.1	20.0
17	1	6.7	17.9	17.4	53.4	50.9	19.6	19.5	19.1	19.0
18	1	5.6	17.9	17.4	53.4	50.9	19.5	19.1	18.6	18.4
19	0	2.8	488.5	267.7	303.7	238.6	19.1	18.0	14.8	9.5
20	0	2.2	488.5	267.7	303.7	238.6	18.0	16.9	13.8	8.7
21	0	1.7	488.5	267.7	303.7	238.6	16.9	15.8	12.8	7.9
22	0	0.6	488.5	267.7	303.7	238.6	15.8	14.7	11.7	6.8
23	0	0.0	488.5	267.7	303.7	238.6	14.7	13.7	10.8	6.0
24	0	-4.4	488.5	267.7	303.7	238.6	13.7	12.4	8.8	3.0
25	0	-5.6	488.5	267.7	303.7	238.6	12.4	11.1	7.6	1.7
26	0	-7.2	488.5	267.7	303.7	238.6	11.1	9.8	6.2	0.3
27	0	-8.9	488.5	267.7	303.7	238.6	9.8	8.5	4.8	-1.3
28	0	-9.4	488.5	267.7	303.7	238.6	8.5	7.2	3.7	-2.1
29	0	-10.6	488.5	267.7	303.7	238.6	7.2	5.9	2.4	-3.3
30	0	-11.7	488.5	267.7	303.7	238.6	5.9	4.7	1.2	-4.5

31	0	-12.2	488.5	267.7	303.7	238.6	4.7	3.5	0.2	-5.3
32	1	-12.2	17.9	17.4	53.4	50.9	3.5	3.1	2.5	2.3
33	1	-12.8	17.9	17.4	53.4	50.9	3.1	2.8	2.2	1.9
34	1	-12.8	17.9	17.4	53.4	50.9	2.8	2.4	1.9	1.7
35	1	-12.8	17.9	17.4	53.4	50.9	2.4	2.2	1.7	1.4
36	1	-13.9	17.9	17.4	53.4	50.9	2.2	1.9	1.3	1.1
37	1	-15.0	17.9	17.4	53.4	50.9	1.9	1.6	1.0	0.7
38	1	-15.0	17.9	17.4	53.4	50.9	1.6	1.2	0.7	0.4
39	1	-15.0	17.9	17.4	53.4	50.9	1.2	0.9	0.4	0.1
40	1	-16.1	17.9	17.4	53.4	50.9	0.9	0.5	-0.1	-0.4
41	1	-17.2	17.9	17.4	53.4	50.9	0.5	0.1	-0.6	-0.9
42	1	-17.8	17.9	17.4	53.4	50.9	0.1	-0.3	-1.0	-1.3
43	0	-17.8	488.5	267.7	303.7	238.6	-0.3	-1.6	-5.0	-10.7
44	0	-18.3	488.5	267.7	303.7	238.6	-1.6	-2.8	-6.0	-11.5
45	0	-18.3	488.5	267.7	303.7	238.6	-2.8	-3.9	-6.9	-11.9
46	0	-18.9	488.5	267.7	303.7	238.6	-3.9	-4.9	-7.9	-12.7
47	0	-19.4	488.5	267.7	303.7	238.6	-4.9	-6.0	-8.8	-13.5
48	0	-18.9	488.5	267.7	303.7	238.6	-6.0	-6.9	-9.4	-13.6

8 Appendix C – Code

C.1

This section shows the Python solver which integrates the matrix inversion methodology, extracting data and coefficients from the hybrid grey-box RC model. Here is an example of it extracting from the adapted cases:

```
# -*- coding: utf-8 -*-
"""
Created on Fri Feb 14 17:47:46 2025

@author: prigm
"""
# -*- coding: utf-8 -*-
"""
Automated Solver with Dynamic A-Matrix and b-Matrix Selection

@author: prigm
```

```
"""
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import os
from matplotlib.lines import Line2D

# The cases selected represent the first that pass all ASHRAE 140
cases = ["600FF1.08.xlsx", "650FF1.08.xlsx", "900FF1.08.xlsx",
"950FF1.08.xlsx"]

def solve_case(file_path, case_name, pass_fail_ranges):
    print(f"Processing case: {case_name}\n{'-' * 40}")

    data = pd.read_excel(file_path, sheet_name="Model_grey-box3")
    coefficients = pd.to_numeric(data.iloc[24:44, 3],
errors='coerce').to_numpy()

    daily_schedule = data["Daily schedule ventilation"].to_numpy()

    #known b matrix
    b_matrix_fixed = data[["Eq1-known term", "Eq2-known
term"]].to_numpy()
    X_solutions = []

    # Iteration through each row assigning the correct a matrix
based on the ventilation schdedule
    # The replacment of the Boolean switching mentioned
    for i in range(len(b_matrix_fixed)):
        if daily_schedule[i] == 1: # Occuppied ventilation
            A_3x3 = np.array([
                [coefficients[9], -coefficients[7], 0],
                [-coefficients[1], coefficients[10], -
coefficients[0]],
                [0, -coefficients[0], coefficients[11]]
            ])
            eq3_term = data["Eq3-known term"].iloc[i]  # Use
occupied ventilation
        else: # Unoccuppied ventilation basically 0
            A_3x3 = np.array([
                [coefficients[9], -coefficients[7], 0],
                [-coefficients[1], coefficients[10], -
coefficients[0]],
                [0, -coefficients[0], coefficients[12]]  # Occupied
ventilation
            ])
            eq3_term = data["Eq3-known term (different
ventilation)"].iloc[i]  # Use different ventilation term

        b_matrix_row = np.array([b_matrix_fixed[i, 0],
b_matrix_fixed[i, 1], eq3_term])

        A_3x3_inverse = np.linalg.inv(A_3x3)

        X_solution = np.dot(A_3x3_inverse, b_matrix_row)
        X_solutions.append(X_solution)

    solution_df = pd.DataFrame(X_solutions, columns=["Theta_m",
"Theta_sup", "Theta_air"])

    # Calculate the values of indoor air temperature from the data
set
    theta_air_values = solution_df["Theta_air"]
```

```
    theta_air_min = theta_air_values.min()
    theta_air_max = theta_air_values.max()
    theta_air_avg = theta_air_values.mean()

    # Apply pass/fail conditions
    min_status = "Pass" if pass_fail_ranges["min"][0] <=
theta_air_min <= pass_fail_ranges["min"][1] else "Fail"
    max_status = "Pass" if pass_fail_ranges["max"][0] <=
theta_air_max <= pass_fail_ranges["max"][1] else "Fail"
    avg_status = "Pass" if pass_fail_ranges["avg"][0] <=
theta_air_avg <= pass_fail_ranges["avg"][1] else "Fail"

    print(f"Summary for {case_name}:")
    print(f"Minimum Theta_air: {theta_air_min:.2f}°C
({min_status})")
    print(f"Maximum Theta_air: {theta_air_max:.2f}°C
({max_status})")
    print(f"Average Theta_air: {theta_air_avg:.2f}°C
({avg_status})")
    print("\n")

    return solution_df

# Pass/fail ranges for each case
pass_fail_ranges = {
    "600FF": {"min": (-18.8, -15.6), "max": (64.9, 75.1), "avg":
(24.2, 27.4)},
    "650FF": {"min": (-23.0, -21), "max": (63.2, 73.5), "avg": (18,
20.8)},
    "900FF": {"min": (-6.4, -1.6), "max": (41.8, 46.4), "avg":
(24.5, 27.5)},
    "950FF": {"min": (-20.2, -17.8), "max": (35.5, 38.5), "avg":
(14.0, 15.3)}
}
for case_file in cases:
    case_name = case_file.replace("1.08.xlsx", "")
    solve_case(case_file, case_name, pass_fail_ranges[case_name])
```

C.2

This section is the code used to carry out the sensitivity analysis in Figure 3.

```
sensitivity_factors = [round(1.00 + i * 0.01, 2) for i in range(11)]
base_cases = ["600FF", "650FF", "900FF", "950FF"]
min_pass_ranges = {
    "600FF": (-18.8, -15.6),
    "650FF": (-23.0, -21.0),
    "900FF": (-6.4, -1.6),
    "950FF": (-20.2, -17.8)
}

shaded_colors = {
    "600FF": "#c6f5c6",
    "650FF": "#a5e4a5",
    "900FF": "#8ddc8d",
    "950FF": "#75d475"
}
```

```
results = {case: [] for case in base_cases}
# Same code and process as listing C.1
def get_min_theta_air(file_path, case):
    data = pd.read_excel(file_path, sheet_name="Model_grey-box3")
    coefficients = pd.to_numeric(data.iloc[24:44, 3],
errors='coerce').to_numpy()
    daily_schedule = data["Daily schedule ventilation"].to_numpy()
    b_matrix_fixed = data[["Eq1-known term", "Eq2-known
term"]].to_numpy()
    X_solutions = []

    for i in range(len(b_matrix_fixed)):
        if daily_schedule[i] == 1:
            A_3x3 = np.array([
                [coefficients[9], -coefficients[7], 0],
                [-coefficients[1], coefficients[10], -
coefficients[0]],
                [0, -coefficients[0], coefficients[11]]
            ])
            eq3_term = data["Eq3-known term"].iloc[i]
        else:
            A_3x3 = np.array([
                [coefficients[9], -coefficients[7], 0],
                [-coefficients[1], coefficients[10], -
coefficients[0]],
                [0, -coefficients[0], coefficients[12]]
            ])
            eq3_term = data["Eq3-known term (different
ventilation)"].iloc[i]

        b_matrix_row = np.array([b_matrix_fixed[i, 0],
b_matrix_fixed[i, 1], eq3_term])
        A_3x3_inverse = np.linalg.inv(A_3x3)
        X_solution = np.dot(A_3x3_inverse, b_matrix_row)
        X_solutions.append(X_solution)

    solution_df = pd.DataFrame(X_solutions, columns=["Theta_m",
"Theta_sup", "Theta_air"])
    return solution_df["Theta_air"].min()

for case in base_cases:
    for factor in sensitivity_factors:
        file_name = f"{case}.xlsx" if factor == 1.00 else
f"{case}{factor:.2f}.xlsx"
        if os.path.exists(file_name):
            min_theta, is_pass = None, False
            try:
                min_theta = get_min_theta_air(file_name, case)
                is_pass = min_pass_ranges[case][0] <= min_theta <=
min_pass_ranges[case][1]
            except Exception as e:
                print(f"Error processing {file_name}: {e}")
            results[case].append((factor, min_theta, is_pass))
        else:
            results[case].append((factor, np.nan, False))
plt.figure(figsize=(10, 6))

for case in base_cases:
    pass_min, pass_max = min_pass_ranges[case]
    plt.axhspan(pass_min, pass_max, color=shaded_colors[case],
alpha=0.2)

    for factor, theta, is_pass in results[case]:
```

```
            if np.isnan(theta):
                continue
            color = "green" if is_pass else "red"
            plt.scatter(factor, theta, color=color, edgecolor='black',
s=60, zorder=5)


        x_vals = [x[0] for x in results[case] if not np.isnan(x[1])]
        y_vals = [x[1] for x in results[case] if not np.isnan(x[1])]
        plt.plot(x_vals, y_vals, marker='o', label=case)

plt.xlabel("Cm/Δτ Increase (%)")
plt.xticks(ticks=sensitivity_factors, labels=[f"{(f -
1.00)*100:.0f}%" for f in sensitivity_factors])
plt.ylabel("Minimum Indoor Air Temperature (°C)")
#plt.title("Sensitivity Analysis of Minimum Theta_air to Cm/Δτ
Variations")
plt.grid(True, linestyle='--', alpha=0.6)
case_lines = [
    Line2D([0], [0], color='tab:blue', lw=2, label='600FF'),
    Line2D([0], [0], color='tab:orange', lw=2, label='650FF'),
    Line2D([0], [0], color='tab:green', lw=2, label='900FF'),
    Line2D([0], [0], color='tab:red', lw=2, label='950FF'),
]

pass_fail_markers = [
    Line2D([0], [0], marker='o', color='green', linestyle='None',
label='Pass'),
    Line2D([0], [0], marker='o', color='red', linestyle='None',
label='Fail')
]

combined_legend = case_lines + pass_fail_markers
plt.legend(handles=combined_legend, title="Case & Status",
loc='center right')
plt.tight_layout()
plt.show()
```

C.3

The code used for the ML regression models was carried out in Jupyter Notebook to install relevant libraries. This list represents the code used to produce the plots of regression for XGBoost and its resdiuals.

```
!pip install xgboost

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import xgboost as xgb
from sklearn.model_selection import train_test_split
from sklearn.metrics import r2_score, mean_absolute_error

cases = ["600FF1.08.xlsx", "650FF1.08.xlsx", "900FF1.08.xlsx",
"950FF1.08.xlsx"]
labels = ["600FF", "650FF", "900FF", "950FF"]

for file, label in zip(cases, labels):
    print(f"\nProcessing case: {label}")
    df = pd.read_excel(file, sheet_name="Model_grey-box3")
```

```
    x = pd.to_numeric(df["te"], errors="coerce")          # Outdoor
temp
    y = pd.to_numeric(df["tair,model"], errors="coerce") # Indoor
temp
    df = pd.DataFrame({"x": x, "y": y}).dropna()
    X_train, X_test, y_train, y_test = train_test_split(
        df["x"].values.reshape(-1, 1), df["y"].values,
test_size=0.2, random_state=42
    )

    dtrain = xgb.DMatrix(X_train, label=y_train)
    dtest = xgb.DMatrix(X_test, label=y_test)
    params = {
        "objective": "reg:squarederror",
        "max_depth": 3,
        "eta": 0.1,
        "verbosity": 0
    }
    model = xgb.train(params, dtrain, num_boost_round=100)
    y_pred = model.predict(dtest)
    r2 = r2_score(y_test, y_pred)
    mae = mean_absolute_error(y_test, y_pred)
    print(f"R²: {r2:.3f}, MAE: {mae:.2f}°C")

    # prediction curve smoothing
    x_range = np.linspace(df["x"].min(), df["x"].max(),
300).reshape(-1, 1)
    dplot = xgb.DMatrix(x_range)
    y_range_pred = model.predict(dplot)

    # Plot 1: Prediction
    plt.figure(figsize=(9, 6))
    plt.scatter(df["x"], df["y"], alpha=0.3, color="steelblue",
label= f"{label} Data Points", s=10)
    plt.plot(x_range, y_range_pred, color="darkred", linewidth=2,
label="XGBoost Prediction")
    plt.text(0.05, 0.95, f"R²: {r2:.3f}\nMAE: {mae:.2f}°C",
         transform=plt.gca().transAxes,
         fontsize=10, verticalalignment='top',
         bbox=dict(boxstyle="round,pad=0.3", facecolor='white',
alpha=0.6))
    plt.xlabel("Outdoor Air Temperature (°C)")
    plt.ylabel("Indoor Air Temperature (°C)")
    plt.legend()
    plt.grid(True, linestyle="--", alpha=0.6)
    plt.tight_layout()
    plt.savefig(f"{label}_XGBoost_Prediction.png", dpi=300)
    plt.show()

    # Plot 2: Residuals
    residuals = y_test - y_pred
    plt.figure(figsize=(8, 4))
    plt.scatter(X_test, residuals, alpha=0.3, color='purple')
    plt.axhline(0, linestyle='--', color='black', linewidth=1)
    #plt.title(f"Residuals: {label} (Indoor - Predicted)")
    plt.xlabel("Outdoor Air Temperature (°C)")
    plt.ylabel("Residuals (°C)")
    plt.grid(True, linestyle='--', alpha=0.5)
    plt.legend()
    plt.tight_layout()
    plt.savefig(f"{label}_XGBoost_Residuals.png", dpi=300)
    plt.show()
```

C.4

This section shows the code for the NN regression model using standard TensorFlow Keras API library. It provides the plots of predictions and residuals.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Input
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import r2_score, mean_absolute_error

cases = ["600FF1.08.xlsx", "650FF1.08.xlsx", "900FF1.08.xlsx",
"950FF1.08.xlsx"]
labels = ["600FF", "650FF", "900FF", "950FF"]

for file, label in zip(cases, labels):
    print(f"Processing case: {label}")
    df = pd.read_excel(file, sheet_name="Model_grey-box3")
    outdoor_temp = pd.to_numeric(df["te"], errors="coerce")  #
Column Q
    indoor_temp = pd.to_numeric(df["tair,model"], errors="coerce")
# Column AB
    valid = ~outdoor_temp.isna() & ~indoor_temp.isna()
    X = outdoor_temp[valid].values.reshape(-1, 1)
    y = indoor_temp[valid].values.reshape(-1, 1)
    scaler_X = StandardScaler()
    scaler_y = StandardScaler()
    X_scaled = scaler_X.fit_transform(X)
    y_scaled = scaler_y.fit_transform(y)

    model = Sequential([
        Input(shape=(1,)),
        Dense(64, activation='relu'),
        Dense(64, activation='relu'),
        Dense(1)
    ])
    model.compile(optimizer='adam', loss='mse')
    model.fit(X_scaled, y_scaled, epochs=100, verbose=0)

    y_pred_scaled = model.predict(X_scaled)
    y_pred = scaler_y.inverse_transform(y_pred_scaled).flatten()
    y_true = y.flatten()

    r2 = r2_score(y_true, y_pred)
    mae = mean_absolute_error(y_true, y_pred)
    print(f"R²: {r2:.3f}, MAE: {mae:.2f}°C")

    sorted_idx = X.flatten().argsort()
    X_sorted = X.flatten()[sorted_idx]
    y_pred_sorted = y_pred[sorted_idx]
    upper_sorted = upper[sorted_idx]
    lower_sorted = lower[sorted_idx]

    # Plot OF PREDICTIONS
    plt.figure(figsize=(10, 6))
    plt.scatter(X, y, alpha=0.3, label=f"{label} Data Points",
color="cornflowerblue", s=10)
```

```
    plt.plot(X_sorted, y_pred_sorted, color='red', linewidth=2,
label="Neural Net Prediction")
    plt.text(0.05, 0.95, f"R²: {r2:.3f}\nMAE: {mae:.2f}°C",
         transform=plt.gca().transAxes,
         fontsize=10, verticalalignment='top',
         bbox=dict(boxstyle="round,pad=0.3", facecolor='white',
alpha=0.6))
    plt.xlabel("Outdoor Air Temperature (°C)")
    plt.ylabel("Indoor Air Temperature (°C)")
    plt.grid(True, linestyle='--', alpha=0.6)
    plt.legend()
    plt.tight_layout()
    plt.savefig(f"{label}_NN_Prediction.png", dpi=300)
    plt.show()

    # Plot of residuals
    plt.figure(figsize=(10, 4))
    plt.scatter(X, residuals, alpha=0.4, color='purple', s=10)
    plt.axhline(0, linestyle='--', color='black', linewidth=1)
    #plt.title(f"Residuals: {label} (Indoor - Predicted)")
    plt.xlabel("Outdoor Air Temperature (°C)")
    plt.ylabel("Residuals (°C)")
    plt.grid(True, linestyle='--', alpha=0.6)
    plt.legend()
    plt.tight_layout()
    plt.savefig(f"{label}_NN_Residuals.png", dpi=300)
    plt.show()
```

C.5

The codes in this section use the same XGBoost ML model but with the added uncertainty visualisation techniques shown in Figures 22-29. The first is the standard deviation envelope:

```
for file, label in zip(cases, labels):
    print(f"\nProcessing case: {label}")

    df = pd.read_excel(file, sheet_name="Model_grey-box3")
    x = pd.to_numeric(df["te"], errors="coerce")
    y = pd.to_numeric(df["tair,model"], errors="coerce")
    df = pd.DataFrame({"x": x, "y": y}).dropna()

    X_train, X_test, y_train, y_test = train_test_split(
        df["x"].values.reshape(-1, 1), df["y"].values,
test_size=0.2, random_state=42
    )
    scaler = StandardScaler()
    X_train_scaled = scaler.fit_transform(X_train)
    X_test_scaled = scaler.transform(X_test)
    X_all_scaled = scaler.transform(df["x"].values.reshape(-1, 1))

    model = xgb.XGBRegressor(n_estimators=100, max_depth=3,
learning_rate=0.1, objective="reg:squarederror", random_state=42)
    model.fit(X_train_scaled, y_train)
    y_pred = model.predict(X_test_scaled)
    r2 = r2_score(y_test, y_pred)
    mae = mean_absolute_error(y_test, y_pred)
    print(f"R²: {r2:.3f}, MAE: {mae:.2f}°C")

    full_pred = model.predict(X_all_scaled)
```

```
    residuals = df["y"].values - full_pred
    std_dev = np.std(residuals)
    upper = full_pred + std_dev
    lower = full_pred - std_dev

    df_sorted = df.copy().reset_index(drop=True).sort_values("x")
    df_sorted["y_pred"] = full_pred
    df_sorted["upper"] = upper
    df_sorted["lower"] = lower
    # Prediction + Standard deviation
    plt.figure(figsize=10, 6)) #to differentiate between the two
groups
    plt.scatter(df_sorted["x"], df_sorted["y"], alpha=0.3,
color="cornflowerblue", s=10, label=f"{label} Data Points")
    plt.plot(df_sorted["x"], df_sorted["y_pred"], color="black",
linewidth=2, label="XGBoost Prediction")
    plt.plot(df_sorted["x"], df_sorted["upper"], color="red",
linestyle="--", alpha=0.6, label="+1 Std Dev")
    plt.plot(df_sorted["x"], df_sorted["lower"], color="red",
linestyle="--", alpha=0.6, label="-1 Std Dev")
    plt.xlabel("Outdoor Air Temperature (°C)")
    plt.ylabel("Indoor Air Temperature (°C)")
    plt.text(0.05, 0.95, f"R²: {r2:.3f}\nMAE: {mae:.2f}°C",
             transform=plt.gca().transAxes,
             fontsize=10, verticalalignment='top',
             bbox=dict(boxstyle="round,pad=0.3", facecolor='white',
alpha=0.6))
    plt.legend()
    plt.grid(True, linestyle='--', alpha=0.6)
    plt.tight_layout()
    plt.savefig(f"{label}_XGBoost_StdDevTube.png", dpi=300)
    plt.show()
```

The second is the quantile envelope between the 10th and 90th percentiles:

```
for file, label in zip(cases, labels):
    print(f"\nProcessing case: {label}")

    df = pd.read_excel(file, sheet_name="Model_grey-box3")
    x = pd.to_numeric(df["te"], errors="coerce")          # Outdoor
    y = pd.to_numeric(df["tair,model"], errors="coerce") # Indoor
    df = pd.DataFrame({"x": x, "y": y}).dropna()
    X_train, X_test, y_train, y_test = train_test_split(
        df["x"].values.reshape(-1, 1), df["y"].values,
test_size=0.2, random_state=42
    )
    scaler = StandardScaler()
    X_train_scaled = scaler.fit_transform(X_train)
    X_test_scaled = scaler.transform(X_test)
    X_all_scaled = scaler.transform(df["x"].values.reshape(-1, 1))

    model = xgb.XGBRegressor(n_estimators=100, max_depth=4,
learning_rate=0.1, random_state=42)
    model.fit(X_train_scaled, y_train)
    y_pred = model.predict(X_test_scaled)
    r2 = r2_score(y_test, y_pred)
    mae = mean_absolute_error(y_test, y_pred)
    print(f"R²: {r2:.3f}, MAE: {mae:.2f}°C")
    x_range = np.linspace(df["x"].min(), df["x"].max(),
300).reshape(-1, 1)
```

```
    x_range_scaled = scaler.transform(x_range)
    y_range_pred = model.predict(x_range_scaled)

    df_sorted = df.sort_values("x").reset_index(drop=True)
    bins = np.linspace(df_sorted["x"].min(), df_sorted["x"].max(),
50)
    df_sorted["bin"] = pd.cut(df_sorted["x"], bins,
include_lowest=True)
    binned = df_sorted.groupby("bin", observed=False).agg({
        "x": "mean",
        "y": [lambda x: np.percentile(x, 10), lambda x:
np.percentile(x, 90)]
    }).dropna()
    binned.columns = ["x_center", "q10", "q90"]
    q10_interp = np.interp(df_sorted["x"], binned["x_center"],
binned["q10"])
    q90_interp = np.interp(df_sorted["x"], binned["x_center"],
binned["q90"])

    # XGBoost + Quantile Tube
    plt.figure(figsize=(8, 6))
    plt.scatter(df_sorted["x"], df_sorted["y"], alpha=0.3,
color='steelblue', s=10, label="Data Points")
    plt.plot(x_range, y_range_pred, color="black", linewidth=2,
label="XGBoost Prediction")
    plt.plot(df_sorted["x"], q10_interp, color="red", linestyle="--
", label="10th Percentile")
    plt.plot(df_sorted["x"], q90_interp, color="red", linestyle="--
", label="90th Percentile")
    plt.xlabel("Outdoor Air Temperature (°C)")
    plt.ylabel("Indoor Air Temperature (°C)")
    #plt.title(f"XGBoost Regression: {label}")
    plt.text(0.05, 0.95, f"R²: {r2:.3f}\nMAE: {mae:.2f}°C",
         transform=plt.gca().transAxes,
         fontsize=10, verticalalignment='top',
         bbox=dict(boxstyle="round,pad=0.3", facecolor='white',
alpha=0.6))
    plt.legend()
    plt.grid(True, linestyle="--", alpha=0.6)
    plt.tight_layout()
    plt.savefig(f"{label}_XGBoost_QuantileTube.png", dpi=300)
    plt.show()
```

9 References

[1] Z. Wang, Y. Hong, L. Huang, M. Zheng, H. Yuan, and R. Zeng, 'A comprehensive review and future research directions of ensemble learning models for predicting building energy consumption', *Energy Build*, vol. 335, p. 115589, May 2025, doi: 10.1016/J.ENBUILD.2025.115589.

[2] International Organization for Standardization (ISO), 'Energy Performance of Buildings—Calculation of Energy Use for Space Heating and Cooling (EN ISO 13790: 2008)', Geneva, 2008.

[3] ASHRAE, 'Standard 140–2017: Standard method of test for the evaluation of building energy analysis computer programs'.

[4] T. Breakthrough Agenda Report, 'The Breakthrough Agenda Report 2024', 2024.

[5] L. Pérez-Lombard, J. Ortiz, and C. Pout, 'A review on buildings energy consumption information', *Energy Build*, vol. 40, no. 3, pp. 394–398, Jan. 2008, doi: 10.1016/J.ENBUILD.2007.03.007.

[6] P. De Wilde, 'The gap between predicted and measured energy performance of buildings: A framework for investigation', *Autom Constr*, vol. 41, pp. 40–49, May 2014, doi: 10.1016/J.AUTCON.2014.02.009.

[7] United Nations Framework Convention on Climate Change (UNFCCC), 'The Paris Agreement'. Accessed: May 01, 2025. [Online]. Available: https://unfccc.int/process-and-meetings/the-paris-agreement

[8] European Commission, '2050 long-term strategy'. Accessed: May 01, 2025. [Online]. Available: https://climate.ec.europa.eu/eu-action/climate-strategies-targets/2050-long-term-strategy_en

[9] P. Westermann and R. Evins, 'Surrogate modelling for sustainable building design – A review', *Energy Build*, vol. 198, pp. 170–186, Sep. 2019, doi: 10.1016/J.ENBUILD.2019.05.057.

[10] United Nations (UN), 'Sustainable Development Goals | United Nations Development Programme'. Accessed: Dec. 05, 2024. [Online]. Available: https://www.undp.org/sustainable-development-goals

[11] V. S. K. V. Harish and A. Kumar, 'A review on modeling and simulation of building energy systems', *Renewable and Sustainable Energy Reviews*, vol. 56, pp. 1272–1292, Apr. 2016, doi: 10.1016/J.RSER.2015.12.040.

[12] L. Tronchin, M. Manfren, and L. C. Tagliabue, 'Optimization of building energy performance by means of multi-scale analysis – Lessons learned from case studies', *Sustain Cities Soc*, vol. 27, pp. 296–306, Nov. 2016, doi: 10.1016/J.SCS.2015.11.003.

[13] D. B. Crawley *et al.*, 'EnergyPlus: creating a new-generation building energy simulation program', *Energy Build*, vol. 33, no. 4, pp. 319–331, Apr. 2001, doi: 10.1016/S0378-7788(00)00114-6.

[14] M. Manfren, P. A. James, and L. Tronchin, 'Data-driven building energy modelling – An analysis of the potential for generalisation through interpretable machine learning', *Renewable and Sustainable Energy Reviews*, vol. 167, p. 112686, Oct. 2022, doi: 10.1016/J.RSER.2022.112686.

[15] Z. Wang and Y. Chen, ‘Data-driven modeling of building thermal dynamics: Methodology and state of the art’, *Energy Build*, vol. 203, p. 109405, Nov. 2019, doi: 10.1016/J.ENBUILD.2019.109405.

[16] C. Miller, ‘More Buildings Make More Generalizable Models—Benchmarking Prediction Methods on Open Electrical Meter Data’, 2019, doi: 10.3390/make1030056.

[17] N. Eslamirad, M. Golamnia, P. Sajadi, and F. Pilla, ‘Leveraging machine learning for data-driven building energy rate prediction’, *Results in Engineering*, vol. 26, p. 104931, Jun. 2025, doi: 10.1016/J.RINENG.2025.104931.

[18] U. Ali *et al.*, ‘Urban building energy performance prediction and retrofit analysis using data-driven machine learning approach’, *Energy Build*, vol. 303, p. 113768, Jan. 2024, doi: 10.1016/J.ENBUILD.2023.113768.

[19] P. Stoffel, M. Berktold, and D. Müller, ‘Real-life data-driven model predictive control for building energy systems comparing different machine learning models’, *Energy Build*, vol. 305, p. 113895, Feb. 2024, doi: 10.1016/J.ENBUILD.2024.113895.

[20] B. Dong, Z. Li, S. M. M. Rahman, and R. Vega, ‘A hybrid model approach for forecasting future residential electricity consumption’, *Energy Build*, vol. 117, pp. 341–351, Apr. 2016, doi: 10.1016/J.ENBUILD.2015.09.033.

[21] M. Brøgger, P. Bacher, and K. B. Wittchen, ‘A hybrid modelling method for improving estimates of the average energy-saving potential of a building stock’, *Energy Build*, vol. 199, pp. 287–296, Sep. 2019, doi: 10.1016/J.ENBUILD.2019.06.054.

[22] P. Michalak, ‘The simple hourly method of EN ISO 13790 standard in Matlab/Simulink: A comparative study for the climatic conditions of Poland’, *Energy*, vol. 75, pp. 568–578, Oct. 2014, doi: 10.1016/J.ENERGY.2014.08.019.

[23] R. Bruno, G. Pizzuti, and N. Arcuri, ‘The Prediction of Thermal Loads in Building by Means of the EN ISO 13790 Dynamic Model: A Comparison with TRNSYS’, *Energy Procedia*, vol. 101, pp. 192–199, Nov. 2016, doi: 10.1016/J.EGYPRO.2016.11.025.

[24] J.-R. Millet, ‘The simple hourly method of prEN 13790: a dynamic method for the future’.

[25] O. T. Ogunsola and L. Song, ‘Review and Evaluation of Using R-C Thermal Modeling of Cooling Load Prediction for HVAC System Control Purpose’, *ASME International Mechanical Engineering Congress and Exposition, Proceedings (IMECE)*, vol. 7, no. PARTS A, B, C, D, pp. 735–743, Oct. 2013, doi: 10.1115/IMECE2012-86988.

[26] M. Raissi, P. Perdikaris, and G. E. Karniadakis, ‘Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations’, *J Comput Phys*, vol. 378, pp. 686–707, Feb. 2019, doi: 10.1016/J.JCP.2018.10.045.

[27] G. E. Karniadakis, I. G. Kevrekidis, L. Lu, P. Perdikaris, S. Wang, and L. Yang, ‘Physics-informed machine learning’, *Nature Reviews Physics 2021 3:6*, vol. 3, no. 6, pp. 422–440, May 2021, doi: 10.1038/s42254-021-00314-5.

[28] Z. Hao *et al.*, ‘Physics-Informed Machine Learning: A Survey on Problems, Methods and Applications’, Nov. 2022. Accessed: Dec. 06, 2024. [Online]. Available: https://arxiv.org/abs/2211.08064v2

[29] X. Chen, T. Guo, M. Kriegel, and P. Geyer, 'A hybrid-model forecasting framework for reducing the building energy performance gap', *Advanced Engineering Informatics*, vol. 52, p. 101627, Apr. 2022, doi: 10.1016/J.AEI.2022.101627.

[30] Y. Gao, Y. Ruan, C. Fang, and S. Yin, 'Deep learning and transfer learning models of energy consumption forecasting for a building with poor information data', *Energy Build*, vol. 223, p. 110156, Sep. 2020, doi: 10.1016/J.ENBUILD.2020.110156.

[31] M. López Santos *et al.*, 'Deep learning and transfer learning techniques applied to short-term load forecasting of data-poor buildings in local energy communities', *Energy Build*, vol. 292, p. 113164, Aug. 2023, doi: 10.1016/J.ENBUILD.2023.113164.

[32] F. Bünning *et al.*, 'Physics-informed linear regression is competitive with two Machine Learning methods in residential building MPC', *Appl Energy*, vol. 310, p. 118491, Mar. 2022, doi: 10.1016/J.APENERGY.2021.118491.

[33] Y. Chen, H. Wang, and Z. Chen, 'Sensitivity analysis of physical regularization in physics-informed neural networks (PINNs) of building thermal modeling', *Build Environ*, vol. 273, p. 112693, Apr. 2025, doi: 10.1016/J.BUILDENV.2025.112693.

[34] Z. Ma, G. Jiang, and J. Chen, 'Physics-informed ensemble learning with residual modeling for enhanced building energy prediction', *Energy Build*, vol. 323, p. 114853, Nov. 2024, doi: 10.1016/J.ENBUILD.2024.114853.

[35] UK Government, 'Home Energy Model: replacement for the Standard Assessment Procedure (SAP)'. Accessed: May 01, 2025. [Online]. Available: https://www.gov.uk/government/consultations/home-energy-model-replacement-for-the-standard-assessment-procedure-sap

[36] L. Lundström, J. Akander, and J. Zambrano, 'Development of a Space Heating Model Suitable for the Automated Model Generation of Existing Multifamily Buildings—A Case Study in Nordic Climate', *Energies 2019, Vol. 12, Page 485*, vol. 12, no. 3, p. 485, Feb. 2019, doi: 10.3390/EN12030485.

[37] M. J. N. Oliveira Panão, C. A. P. Santos, N. M. Mateus, and G. Carrilho Da Graça, 'Validation of a lumped RC model for thermal simulation of a double skin natural and mechanical ventilated test cell', *Energy Build*, vol. 121, pp. 92–103, Jun. 2016, doi: 10.1016/J.ENBUILD.2016.03.054.

[38] P. Michalak, 'The development and validation of the linear time varying Simulink-based model for the dynamic simulation of the thermal performance of buildings', *Energy Build*, vol. 141, pp. 333–340, Apr. 2017, doi: 10.1016/J.ENBUILD.2017.02.047.

[39] Y. Li, Z. O'Neill, L. Zhang, J. Chen, P. Im, and J. DeGraw, 'Grey-box modeling and application for building energy simulations - A critical review', *Renewable and Sustainable Energy Reviews*, vol. 146, p. 111174, Aug. 2021, doi: 10.1016/J.RSER.2021.111174.

[40] G. S. Pavlak, A. R. Florita, G. P. Henze, and B. Rajagopalan, 'Comparison of Traditional and Bayesian Calibration Techniques for Gray-Box Modeling', *Journal of Architectural Engineering*, vol. 20, no. 2, p. 04013011, Dec. 2013, doi: 10.1061/(ASCE)AE.1943-5568.0000145.

[41] A. Zarrella, E. Prataviera, P. Romano, L. Carnieletto, and J. Vivian, 'Analysis and application of a lumped-capacitance model for urban building energy modelling', *Sustain Cities Soc*, vol. 63, p. 102450, Dec. 2020, doi: 10.1016/J.SCS.2020.102450.

[42] International Organization for Standardization (ISO), ‘Energy Performance of Buildings—Calculation of Energy Use for Space Heating and Cooling (EN ISO 13792: 2012)’, Geneva, 2012.

[43] British Standards Institution (BSI), ‘Energy performance of buildings. Energy needs for heating and cooling, internal temperatures and sensible and latent heat loads - Calculation procedures (BS EN ISO 52016-1:2017)’, Geneva, 2017.

[44] M. H. Lin, J. G. Carlsson, D. Ge, J. Shi, and J. F. Tsai, ‘A Review of Piecewise Linearization Methods’, *Math Probl Eng*, vol. 2013, no. 1, p. 101376, Jan. 2013, doi: 10.1155/2013/101376.

[45] XGBoost Developers, ‘XGBoost Documentation — xgboost 3.0.0 documentation’. Accessed: May 01, 2025. [Online]. Available: https://xgboost.readthedocs.io/en/release_3.0.0/

[46] Google Brain, ‘TensorFlow’. Accessed: May 01, 2025. [Online]. Available: https://www.tensorflow.org/

[47] TensorFlow Core Team, ‘Keras: The high-level API for TensorFlow | TensorFlow Core’. Accessed: May 01, 2025. [Online]. Available: https://www.tensorflow.org/guide/keras

[48] Keras Team, ‘Keras: Deep Learning for humans’. Accessed: May 01, 2025. [Online]. Available: https://keras.io/

[49] scikit-learn Developers, ‘StandardScaler — scikit-learn 1.6.1 documentation’. Accessed: May 01, 2025. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html

[50] Keras Team, ‘Adam Optimizer API Documentation’. Accessed: May 01, 2025. [Online]. Available: https://keras.io/api/optimizers/adam/

[51] Keras Team, ‘ReLU layer’. Accessed: May 01, 2025. [Online]. Available: https://keras.io/api/layers/activation_layers/relu/

[52] T. Januschowski, Y. Wang, K. Torkkola, T. Erkkilä, H. Hasson, and J. Gasthaus, ‘Forecasting with trees’, *Int J Forecast*, vol. 38, no. 4, pp. 1473–1481, Oct. 2022, doi: 10.1016/J.IJFORECAST.2021.10.004.

[53] Z. Jiang, X. Wang, and B. Dong, ‘Physics-informed Modularized Neural Network for Advanced Building Control by Deep Reinforcement Learning’.

[54] M. H. Mehraban, S. M. Sepasgozar, A. Ghomimoghadam, and B. Zafari, ‘AI-enhanced automation of building energy optimization using a hybrid stacked model and genetic algorithms: Experiments with seven machine learning techniques and a deep neural network’, *Results in Engineering*, vol. 26, p. 104994, Jun. 2025, doi: 10.1016/J.RINENG.2025.104994.

[55] X. J. Luo, L. O. Oyedele, A. O. Ajayi, O. O. Akinade, H. A. Owolabi, and A. Ahmed, ‘Feature extraction and genetic algorithm enhanced adaptive deep neural network for energy consumption prediction in buildings’, *Renewable and Sustainable Energy Reviews*, vol. 131, p. 109980, Oct. 2020, doi: 10.1016/J.RSER.2020.109980.

[56] B. Jiang, Y. Li, Y. Rezgui, C. Zhang, P. Wang, and T. Zhao, ‘Multi-source domain generalization deep neural network model for predicting energy consumption in multiple office buildings’, *Energy*, vol. 299, p. 131467, Jul. 2024, doi: 10.1016/J.ENERGY.2024.131467.

[57] R. Akter, M. G. Shirkoohi, J. Wang, and W. Mérida, ‘An efficient hybrid deep neural network model for multi-horizon forecasting of power loads in academic

buildings', *Energy Build*, vol. 329, p. 115217, Feb. 2025, doi: 10.1016/J.ENBUILD.2024.115217.