

AUTOMATED IDENTIFICATION OF GRAY-BOX MODELS FOR HVAC SYSTEMS

JOSÉ ANTONIO BUENO UCETA

Master's thesis
2025:E49



LUND UNIVERSITY

Faculty of Science
Centre for Mathematical Sciences
Computational Science

Abstract

Today, buildings account for approximately 30% of worldwide energy use, with HVAC (Heating, Ventilation, and Air Conditioning) systems representing nearly 40% of that total. Improving the energy efficiency of HVAC systems is therefore essential for reducing operational costs and mitigating environmental impact.

Because HVAC systems operate continuously and under strict operational constraints, accurate models are required to evaluate and optimize different operating strategies. Traditional physics-based models (white-box) are often costly and time-consuming to develop, while purely data-driven models (black-box) tend to lack interpretability and robustness. This thesis investigates whether hybrid approaches, referred to as gray-box models, can offer a more efficient and reliable alternative by integrating both physical knowledge and data.

To this end, four modeling strategies are developed and compared: black-box and gray-box models, each implemented with and without a component-based architecture. All models are constructed using Neural Ordinary Differential Equations (NODEs). The goal is to evaluate whether combining physics knowledge with data-driven techniques can reduce the need for large and highly variable training datasets while maintaining or improving model accuracy. The comparison is performed using two key performance metrics: the Relative Root Mean Squared Error (RRMSE) and the Maximum Absolute Error (MAE). Additionally, the complexity required to achieve comparable predictive performance across strategies is analyzed.

The results show that, when sufficient training data is available, gray-box models achieve comparable or superior accuracy with fewer parameters than black-box models. Moreover, the inclusion of physical knowledge improves both predictive accuracy and model robustness in scenarios with limited or low-quality training data. Nevertheless, gray-box models still require datasets with sufficient excitation to ensure stable and reliable performance.

Popular science description

Imagine walking into a building on a hot summer day and feeling a refreshing cool breeze. Most people do not think much about what is happening behind the scenes, but there is a complex system of machines that work continuously to keep the indoor environment comfortable. These are known as HVAC (Heating, Ventilation, and Air Conditioning) systems, and they are used not only for human comfort, but also for industrial purposes such as cooling data centers. Nowadays, HVAC systems account for approximately 10% of the energy consumption in buildings across the globe. Therefore, improving their efficiency can lead to substantial savings in both energy and cost.

Traditionally, engineers have modeled HVAC systems using a “white-box” approach, which relies on physical laws such as thermodynamics and fluid dynamics. These models can be very accurate, but often require significant time and expert knowledge. In the past years, a new approach known as “black-box” uses machine learning techniques to find patterns in data without considering the underlying physics. While easier to construct, black-box models usually yield unreliable predictions outside the range of the data they have been trained on.

In this work, a third alternative is explored, the “gray-box” model, which aims to combine the strengths of both white-box and black-box methods. For this, it uses data from real or simulated systems while also incorporating basic physical knowledge to guide the learning process. The key method used in this thesis is called Neural Ordinary Differential Equations, or NODEs.

Such a model can be used to explore how changes in the operation of individual components influence the system’s performance. Therefore, it is possible to identify configurations that reduce energy consumption without performing lengthy or risky experiments on the actual equipment. This is particularly valuable in environments like data centers, where stability and reliability are crucial.

Acknowledgements

I would like to express my sincere gratitude to my academic supervisor, Philipp Birken, for his continuous guidance, encouragement, and valuable feedback throughout the development of this thesis.

I am also thankful to my supervisor from Carrier, Viktor Linders, whose insights and support were greatly helpful during the initial stages of the project. I would also like to thank Johan Åkesson, who took over the supervision with great commitment and provided valuable assistance during the remainder of the work.

I extend my thanks to the team at Pacific Northwest National Laboratory (PNNL) for their support during the thesis work, particularly for their help in resolving issues related to the NeuroMANCER framework.

I would like to thank my colleagues Simon Horovitz and Oskar Hübsch for their collaboration, constructive discussions, and support throughout this journey.

Finally, I am profoundly grateful to my family and friends for their unwavering support, patience, and encouragement during my studies.

Contents

Popular science description	iii
1 Introduction	1
1.1 Chiller Plants	2
1.2 Modeling Strategies	6
1.3 Previous Work	7
1.4 Objectives	8
1.5 Scope of the Thesis	9
2 Theoretical Background	11
2.1 Multi-Layer Perceptron	11
2.2 Neural Ordinary Differential Equations	14
3 System Description and Modeling Framework	19
3.1 Modeling Tools	19
3.2 System of Study	19
3.3 System Modeling Equations	21
4 Methodology	25
4.1 Dataset Generation	25
4.2 Neural Modeling	26
4.3 Model Training	31
4.4 Model Testing	32
4.5 Set up for the Numerical Experiments	33
5 Numerical Results	35
5.1 Two-Chiller System	35
5.2 Complete HVAC System	39
6 Conclusions	61
7 Future Work	63
8 Appendix	65
9 Bibliography	69

Chapter 1

Introduction

Over the past century, global energy consumption has risen sharply, as shown in [Figure 1.1](#). This increase has been largely driven by population growth and technological advances. In particular, over the last decade, the trend has accelerated even further with the expansion of industries based on big data and machine learning, which heavily rely on data centers to store information and run computational models.

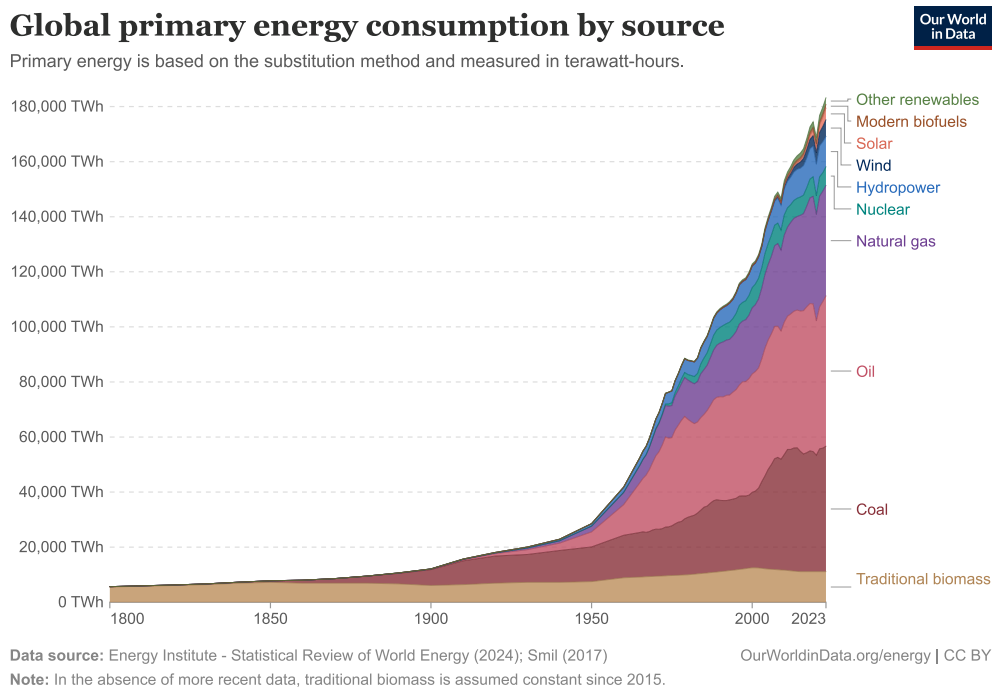


Figure 1.1: Evolution of the energy consumed worldwide per year. Source: [1].

According to the Global Alliance of Buildings and Construction, buildings accounted for approximately 30% of global energy consumption in 2020 [2]. Notably, nearly 40% of this consumption is attributed to heating, ventilation, and air conditioning (HVAC) systems, as detailed in [3]. These are essential for maintaining controlled indoor environments, operating continuously to cool, heat, and ventilate spaces for industrial processes,

commercial activities, and personal comfort. Their role has become increasingly important in modern society, where many environments require tight control over temperature, humidity, and air quality to ensure health, productivity, and the proper functioning of equipment.

As living standards rise and technological development advances, the demand for reliable HVAC systems is projected to grow. This trend is further reinforced by global warming, with average global temperatures expected to exceed 2°C above pre-industrial levels by 2100 [4]. This increasing demand places further strain on global energy resources, emphasizing the need for more energy-efficient solutions.

Moreover, a significant portion of the current building stock, including many data centers, was constructed decades ago, when energy efficiency standards were far less stringent than they are today. For instance, the IEA reports in [5] that in developed economies, at least 40% of the building floor area was constructed before 1980, before the introduction of the first thermal regulations. Nowadays, many of these facilities need to renovate their infrastructure to meet modern demands. This includes not only upgrading IT equipment but also modernizing HVAC systems, which often represent a major portion of operational energy costs.

Optimized HVAC operation can provide energy savings of 25% while maintaining a satisfactory indoor environment, therefore lowering greenhouse gas emissions and reducing costs for building operators [6]. Enhancing the performance and efficiency of HVAC systems thus represents a substantial opportunity for energy and cost savings. As such, the search for smarter, more efficient ways to manage HVAC systems has become a key focus area for both industry and research.

This thesis focuses on chiller plants for data centers and is carried out in collaboration with Carrier, the world's largest provider of HVAC solutions, which recently opened a new office in Lund, Sweden. One of the main goals of this new site is to develop advanced models for optimizing the energy consumption of HVAC systems in data centers. This collaboration provides a unique opportunity to tackle real-world challenges in a rapidly growing and energy-intensive industry.

1.1 Chiller Plants

1.1.1 Chiller

The purpose of a chiller plant is to provide cooling for large buildings or facilities by transferring heat from the building to the environment. Chiller plants are typically categorized by the type of fluid they use: air or water. This project focuses on water-cooled chiller plants, which generally consist of two water loops and one or more air loops. In particular, only the water loops are modeled in this study, as optimizing the air loops involves a higher operational risk.

Figure 1.2 illustrates the chiller [7], a key component responsible for removing heat from the chilled water (blue) returning from the data center and transferring it to the condenser water loop (orange). This continuous heat removal process keeps the chilled water temperature low enough to meet the cooling demands of the facility. The system

operates using a working fluid called refrigerant, which cycles through four main components cyclically, experiencing phase changes and pressure variations to absorb and reject heat, thereby enabling the cooling process.

The cycle begins in the evaporator, where the warm chilled water, having absorbed heat from the IT room air, flows through a heat exchanger in close contact with the refrigerant. The refrigerant, initially at a low pressure and low temperature, absorbs the thermal energy from the water and evaporates.

The low-pressure refrigerant vapor is drawn into the compressor, which performs mechanical work on the vapor, significantly raising its pressure and temperature. This step is vital because it increases the refrigerant's energy, enabling it to transfer the heat to the condenser water.

The high-pressure, high-temperature vapor then enters the condenser, where the refrigerant releases its absorbed heat to the condenser water loop through another heat exchanger. As the refrigerant relinquishes its thermal energy, it condenses back into a high-pressure liquid.

Following condensation, the refrigerant passes through the expansion valve, suffering a sharp drop in pressure and temperature, preparing the refrigerant to absorb heat once again. This low-pressure, low-temperature liquid refrigerant then re-enters the evaporator, completing the cycle.

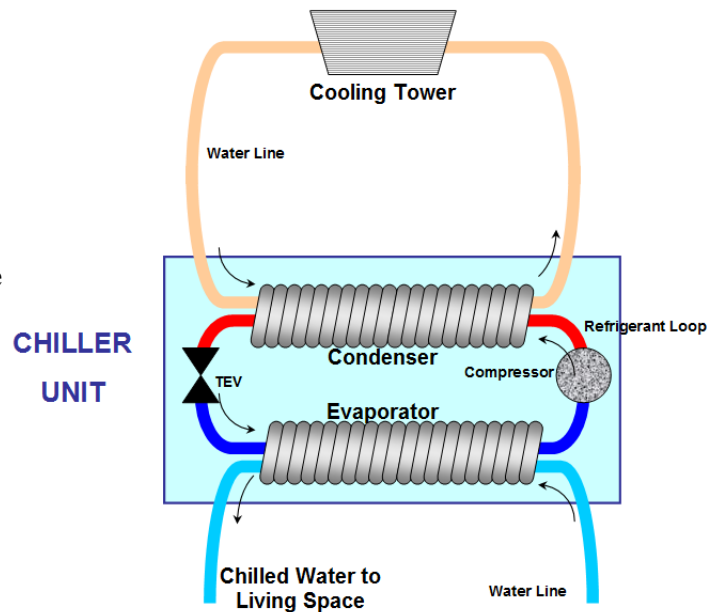


Figure 1.2: Schematic of a chiller system showing the refrigeration cycle components: compressor, condenser, thermal expansion valve (TEV), and evaporator. The interaction between the chiller unit and the condenser water loop (orange) and chilled water loop (blue) is also depicted.

1.1.2 Complete System

While the chiller plays a central role in the refrigeration cycle, the cooling process cannot be achieved by the chiller alone. Additional components are necessary to manage the substantial heat generated by servers and IT equipment and to maintain suitable environmental conditions. Without adequate temperature and humidity control, hardware components are prone to overheating, leading to increased failure rates, reduced performance, and potential data loss.

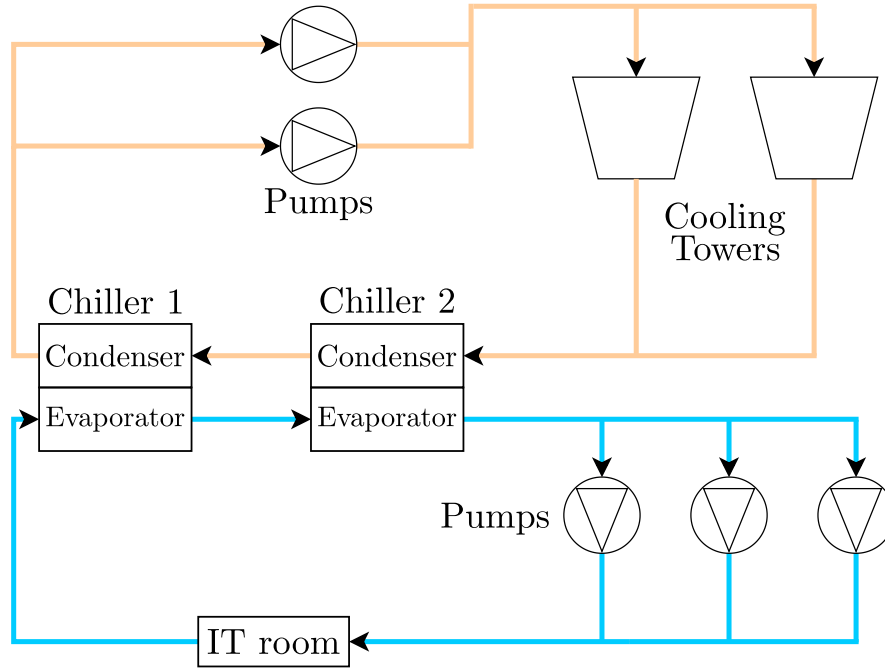


Figure 1.3: Diagram of a chiller plant, depicting the condenser water loop (orange) and the chilled water loop (blue). The system comprises cooling towers, pumps, and two chillers that supply conditioned water to maintain appropriate thermal conditions in the IT room.

Figure 1.3 illustrates the cooling process, which begins when the IT equipment operates and releases heat into the surrounding air, causing the temperature of the room to rise. The warm air from the server rooms is drawn across cooling coils filled with chilled water due to the fans in the Air Handling Unit (AHU). As the air passes over the coils, heat is transferred from the air to the chilled water, cooling the air before it is recirculated back into the room to maintain the desired conditions.

After absorbing heat, the chilled water flows to the chillers at a higher temperature, driven by pumps that guarantee a continuous flow capable of transporting the extracted heat. Inside the chillers, the heat carried by the chilled water is transferred to the condenser water loop, as explained in the previous section. This allows the chilled water to cool down before being sent back to the cooling coils to repeat the cycle.

The cooling water, now warmed by the transferred heat, is similarly circulated by pumps toward the cooling towers. There, the absorbed heat is expelled into the atmosphere, usually through the evaporation of water, and the cooled water returns to the chillers to close the loop.

Therefore, the overall operation forms a closed-loop system where heat generated by servers is first transferred to the air, then extracted by chilled water in the cooling coils, transported to the chillers, passed onto the cooling water, and finally expelled into the environment through the cooling towers.

1.1.3 Building Management System

The Building Management System (BMS) plays a vital role in ensuring both the operational efficiency and the safety of the components within a chiller plant. It achieves this by collecting real-time measurements from sensors distributed throughout the system and executing control strategies that regulate the operation of plant equipment. A key function of the BMS is to ensure that dynamic variables, such as water temperature at different locations in the circuits, closely track predefined setpoints. In real-world applications, these setpoints are dynamically adjusted by a supervisory control layer to enhance overall system efficiency in response to changing conditions.

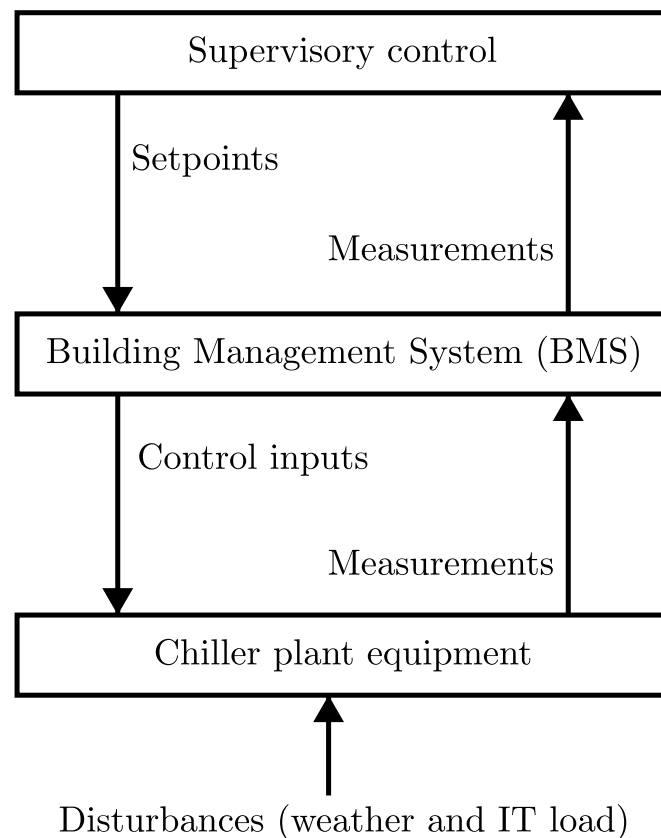


Figure 1.4: Interaction between the supervisory control layer, the Building Management System (BMS), and the chiller plant equipment. The BMS receives setpoints from a higher-level control layer and translates them into control inputs for the equipment, while also collecting measurements to inform future decisions.

Figure 1.4 illustrates the hierarchical control structure typically found in a chiller plant. At the top, a supervisory control layer determines optimal setpoints based on broader system objectives, such as energy efficiency or load balancing. These setpoints are passed to the Building Management System (BMS), which acts as an intermediary between high-level control logic and physical equipment. The BMS translates setpoints into actionable control inputs and dispatches them to the chiller plant equipment. In return, the BMS collects measurements from sensors embedded in the equipment, which

are used to monitor performance and adjust control actions. The system is also subject to external disturbances, such as weather conditions or variable IT loads, which influence the thermal demand and must be considered when analyzing system dynamics.

In this project, we focus solely on modeling the combination of the BMS and the chiller plant equipment. The supervisory control layer is excluded from the analysis. Consequently, the setpoints are treated as predefined control inputs whose time evolution is specified for each experimental scenario.

1.2 Modeling Strategies

Several modeling strategies can be adopted for modeling a chiller plant, which can be divided into three classes according to how much physics or data they use [8]. These are:

- White-box model. They are explicitly derived from first-principles physics, such as thermodynamics, fluid dynamics, and heat transfer. These models offer strong interpretability and generalizability under unseen conditions, as their structure reflects the underlying physics of the system.

However, developing such models is often time-consuming, since it requires a deep understanding of each component's behavior. Moreover, real systems exhibit complex interactions and nonlinearities that are often difficult or impossible to capture explicitly.

- Black-box model. They are based on purely data-driven techniques, such as neural networks (NNs), to learn system behavior directly from data. This eliminates the need for detailed physical modeling and allows the capture of complex system behaviors that may be difficult to describe analytically.

While this approach reduces modeling effort, it relies on large and diverse datasets to generalize well. In HVAC applications, however, such data is often scarce or lacks sufficient variability, as systems typically operate within narrow, safe ranges to avoid compromising equipment performance and longevity.

- Gray-box model. This approach combines physics-based modeling with data-driven techniques. Gray-box models aim to balance the interpretability and generalization of physics-based models with the adaptability of machine learning. However, they still require system analysis to define the baseline equations and sufficiently rich datasets to train the neural networks. The modeling objective is thus to leverage existing physical knowledge while using data-driven techniques to capture unmodeled dynamics.

In this project, two modeling approaches are considered for constructing gray- and black-box models, based on how neural networks are applied:

- Non-component-based. A single neural network is used to model the entire plant. This simplifies model design and eliminates the need to specify subsystem boundaries. However, it requires a larger number of parameters and loses interpretability

compared to the component-based approach. This arises because, for example, the temperature of the water leaving the cooling towers is not directly influenced by the temperature of the water leaving the IT room. As a result, the model must infer this lack of dependency solely from the data, increasing the complexity of the learning task.

- **Component-based.** Each subsystem of the plant, such as chillers, pumps, and cooling towers, is modeled using a separate neural network. This modular structure reduces the learning task on individual networks, improves interpretability, and typically requires fewer parameters overall. This is because physically unrelated variables are naturally separated by using different neural networks. However, it also makes it more difficult to design the model, as it demands knowledge of the system architecture.

1.3 Previous Work

Modeling heating, ventilation, and air conditioning (HVAC) systems is essential for accurate energy consumption estimation, control design, and predictive maintenance. As mentioned previously, the main approaches to HVAC modeling can be categorized into data-driven, physics-based, and gray-box methods. This section reviews significant contributions in each category, highlighting some of the methods used to model HVAC systems [8].

Data-driven approaches, also known as black-box models, construct input–output relationships directly from empirical data without incorporating prior physical knowledge. One widely used technique is the frequency domain model with dead time, which captures the slow dynamics and delays characteristic of HVAC systems, particularly in zone temperature and humidity regulation. These models, typically expressed as first- or second-order transfer functions, are simple, require few parameters, and support straightforward controller design. However, their applicability is limited to linear, time-invariant systems and requires controlled excitation for parameter identification [9–12].

Another major category comprises data mining algorithms, such as artificial neural networks (ANNs) and support vector machines (SVMs), which are capable of modeling complex nonlinear behaviors from large datasets. These models have been successfully applied in tasks ranging from load forecasting [13] to fault detection [14] and energy consumption prediction [15]. While they achieve high accuracy, their dependence on extensive training data and lack of physical interpretability limit their generalization and real-time applicability.

Statistical models like autoregressive with exogenous input (ARX), ARMAX, and ARIMA provide a structured framework for capturing temporal dependencies in HVAC processes. These models are computationally efficient, work well under time-varying noise, and are particularly useful in control-oriented applications. However, their performance degrades in the presence of strong nonlinearities and they often require extensive parameter tuning for multi-input multi-output (MIMO) systems [16].

In contrast to black-box models, physics-based models are derived from fundamental thermodynamic and fluid dynamic principles and are typically designed to represent the behavior of specific HVAC components. The chiller model, for instance, is often constructed by modeling its constituent subsystems: evaporator, condenser, compressor, and expansion valve. These components operate under a vapor compression cycle [17], and their behavior can be described using energy balance equations and thermodynamic relationships. The chiller power consumption is modeled as a function of water flow rates, temperature differentials, and the coefficient of performance (COP) [18], which itself depends on the part load ratio (PLR) and operating conditions [19].

The cooling tower model represents the operation of this subsystem, which serves to reject heat from the chiller condenser. Physics-based models are commonly developed using Merkel’s method or the Effectiveness–NTU approach, while the empirical model is based on Stoecker’s method [20].

Fans and pumps are modeled to quantify both the flow rates they induce and the energy they consume. The models relate flow rates to pressure differences and account for mechanical and electrical efficiencies. The power consumption of the fan or pump depends on the flow rate, pressure difference between inlet and outlet and efficiency of the fan or pump [18, 21, 22].

Finally, gray-box models use physics-based models as the model structure, whose parameters are estimated from the measured data. These models provide physical meaning and are useful for control applications with generalization capability superior to data-driven models. A common approach is to formulate thermal zone dynamics using RC networks, with parameters such as thermal resistances and capacitances estimated via optimization algorithms like genetic algorithms (GA)[23], nonlinear least squares, or sequential quadratic programming [24]. Such models have demonstrated accurate predictions over extended time horizons and robustness across varying conditions.

1.4 Objectives

This thesis investigates the use of Neural Ordinary Differential Equations (NODEs) to model the dynamic behavior of a chiller plant in combination with its Building Management System (BMS). NODEs offer a continuous-time modeling framework that can be applied to both purely data-driven and physics-informed models.

The first objective of this work is to assess whether NODEs can be used to develop accurate and reliable models of the integrated system composed of the BMS and the chiller plant. Good models are essential for predicting the system behavior in different scenarios and supporting the design of advanced control strategies for energy efficiency and safety.

The second objective is to determine whether the inclusion of physical knowledge leads to improved performance in terms of accuracy, generalization, data efficiency, and model complexity. To address these goals, four experiments were conducted, each designed to test a specific hypothesis.

In the first experiment, the focus was on the model structure. The aim was to compare the performance of the different modeling approaches when there is enough training

data with enough variability. It was hypothesized that, yielding similar prediction accuracy, physics-informed models would require fewer parameters than purely data-driven models, and that component-based models would also be more memory-efficient than non-component-based ones. The results confirmed this hypothesis: physics-informed models, by leveraging prior physical knowledge, achieved accurate results with fewer parameters. Similarly, component-based models require fewer parameters because the model structure already accounts for the lack of relationship between physically independent variables.

The second experiment examined how the models behave when the amount of training data is limited. It was hypothesized that physics-informed models, guided by prior knowledge, would be more robust and stable under limited data conditions than their data-driven counterparts. This was supported by the results, which showed that physics-informed models maintained higher overall accuracy and stability on unseen data when less training data was available.

The third experiment assessed the ability of each model to extrapolate beyond the training data. Models were trained on a restricted operational range and tested on scenarios outside this range. The hypothesis was that models incorporating physical knowledge would generalize better due to the incorporated physics-based knowledge. The results confirmed this: physics-informed models demonstrated better extrapolation performance, while purely data-driven models often produced unstable predictions.

The fourth experiment investigated model performance when the training data exhibits low variability, i.e., when the system is only weakly excited. The hypothesis was that physics-informed models would still be able to capture the system dynamics under such conditions. However, the results showed that both physics-informed and purely data-driven models performed poorly and were often unstable when trained on low-excitation data. This suggests that, despite the advantages of embedding physics, sufficient excitation in the input signals is essential for reliably learning system behavior.

1.5 Scope of the Thesis

As previously mentioned, this thesis focuses on the development and evaluation of gray-box and black-box models of a chiller plant using neural ordinary differential equations (NODEs) for different scenarios. To do this, the modeling approach involves assumptions and simplifications, outlined below.

First, discrete variables, such as the on/off status of components, are not considered. Second, transport delays caused by the piping network are not included in the model. These decisions are motivated by the fact that the NODE implementation framework used in this thesis does not support the modeling of discrete dynamics or varying time delays. In practice, all components of the chiller plant are typically active during summer and high-load conditions, particularly when the IT load is higher than 50% of the maximum capacity.

Finally, the data used for training, validating and testing the gray-box and black-box models is generated from a simulation of the chiller plant. Consequently, the measurements are free of noise and other uncertainties typically present in real sensor data.

Chapter 2

Theoretical Background

2.1 Multi-Layer Perceptron

In this report, feedforward Multi-Layer Perceptrons will be used to construct the black and gray-box models. This structure comprises multiple layers of neurons, where information propagates sequentially from the input to the output without feedback loops.

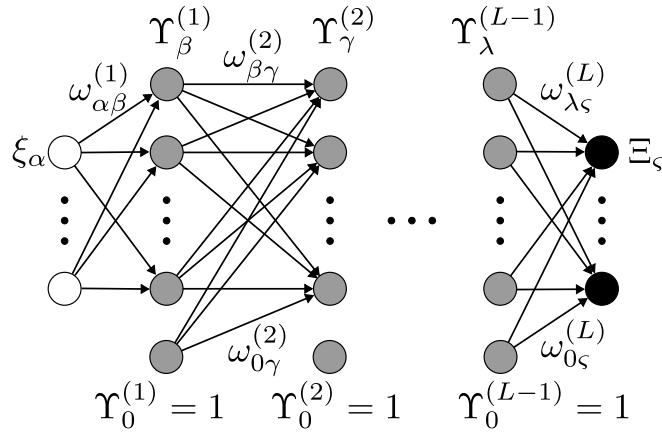


Figure 2.1: Schematic representation of a feedforward fully connected multilayer perceptron (MLP). The white circles represent the input layer nodes, while the black circles correspond to the output layer nodes. The hidden layer nodes are depicted as gray circles, the first one being the bias. The arrows indicate the connections between nodes and the direction of information flow through the network.

Figure 2.1 shows the general structure of an MLP, which consists of the input layer, one or more hidden layers, and the output layer. In this work, we employ fully connected networks, meaning that each neuron in a given layer is connected to all neurons in the subsequent layer. These connections are associated with adjustable weights (ω) that determine the strength of the connections between neurons.

The network receives the input vector $\xi = [\xi_\alpha]_{\alpha=1}^{m_0} \in \mathbb{R}^{m_0}$, where each element corresponds to a node in the input layer, which forwards its value to the first hidden layer. There, each node applies a non-polynomial activation function φ to a weighted sum of

the inputs:

$$\Upsilon_{\beta}^{(1)} = \varphi \left(\sum_{\alpha=1}^{m_0} \omega_{\alpha\beta}^{(1)} \xi_{\alpha} \right) = \varphi \left(\boldsymbol{\omega}_{\beta}^{(1)T} \boldsymbol{\xi} \right),$$

where $\omega_{\alpha\beta}^{(1)}$ is the weight connecting the α -th neuron in the input layer to the β -th neuron in the first hidden layer. In vector form for all nodes:

$$\boldsymbol{\Upsilon}^{(1)} = \varphi \left(\mathbf{W}^{(1)T} \boldsymbol{\xi} \right),$$

where $\mathbf{W}^{(1)} = \left[\boldsymbol{\omega}_{\beta}^{(1)} \right]_{\beta=1}^{m_1}$ is the weight matrix of the first hidden layer and the activation function φ is applied element-wise. In this project, it is the same for all hidden and output nodes. In particular, we will use the Exponential Linear Unit (ELU) and the Rectified Linear Unit (ReLU), which are given by

$$\text{ELU}(x) = \begin{cases} x & , \text{ if } x \geq 0, \\ \alpha(e^x - 1) & , \text{ if } x < 0, \end{cases} \quad \text{and} \quad \text{ReLU}(x) = \max(0, x),$$

where $\alpha > 0$ is a hyperparameter that controls the value to which an ELU saturates for negative inputs.

Next, the output of the first hidden layer is sent to the second one, which applies a similar transformation. Additionally, the bias vector $\boldsymbol{\omega}_0^{(2)}$ is introduced to allow for shifts in the argument of the activation function, improving the flexibility of the neural network. Hence,

$$\boldsymbol{\Upsilon}^{(2)} = \varphi \left(\mathbf{W}^{(2)T} \boldsymbol{\Upsilon}^{(1)} + \boldsymbol{\omega}_0^{(2)} \right) = \varphi \left(\mathbf{W}^{(2)T} \varphi \left(\mathbf{W}^{(1)T} \boldsymbol{\xi} \right) + \boldsymbol{\omega}_0^{(2)} \right).$$

This process is repeated for the consecutive hidden layers. Therefore, the expression corresponding to the k :th layer is

$$\boldsymbol{\Upsilon}^{(k)} = \varphi \left(\mathbf{W}^{(k)T} \boldsymbol{\Upsilon}^{(k-1)} + \boldsymbol{\omega}_0^{(k)} \right),$$

for $k = 2, \dots, L-1$.

Finally, the output layer produces the result of the network $\boldsymbol{\Xi}$, which is given by

$$\boldsymbol{\Xi} = \varphi \left(\mathbf{W}^{(L)T} \boldsymbol{\Upsilon}^{(L-1)} + \boldsymbol{\omega}_0^{(L)} \right)$$

2.1.1 Training Process

Let f be an unknown function such that

$$\mathbf{y} = f(\mathbf{x})$$

where $\mathbf{x} \in \mathbb{R}^{n_x}$ and $\mathbf{y} \in \mathbb{R}^{n_y}$. Assume access to a training dataset of $N \in \mathbb{N}$ measurements, organized as column-wise matrices $\hat{\mathbf{x}} = [\hat{\mathbf{x}}_j]_{j=0}^{N-1} \in \mathbb{R}^{n_x \times N}$ and $\hat{\mathbf{y}} = [\hat{\mathbf{y}}_j]_{j=0}^{N-1} \in \mathbb{R}^{n_y \times N}$.

Let $\mathcal{N}(\mathbf{x}; \boldsymbol{\omega})$ denote a multilayer perceptron (MLP) with weights and biases $\boldsymbol{\omega}$. Training the MLP aims to find the optimal parameters $\boldsymbol{\omega}^*$ that minimize a loss function $\mathcal{L}(\boldsymbol{\omega})$, which typically quantifies the difference between the predicted output $\mathbf{y} = \mathcal{N}(\mathbf{x}; \boldsymbol{\omega})$ and the target $\hat{\mathbf{y}}$:

$$\min_{\boldsymbol{\omega}} \mathcal{L}(\boldsymbol{\omega})$$

This optimization is generally performed using gradient-based algorithms that require the computation of $\nabla_{\omega}\mathcal{L}(\omega)$ [25]. In this work, we employ the Adaptive Moment Estimation (ADAM) optimizer [26], described in Algorithm 1.

Algorithm 1 The Adam algorithm

Require: Learning rate ϵ ($\epsilon = 0.001$ in this work)

Require: Exponential decay rates for moment estimates, ρ_1 and ρ_2 in $[0, 1]$ ($\rho_1 = 0.9$ and $\rho_2 = 0.999$ in this work)

Require: Small constant δ used for numerical stabilization ($\delta = 10^{-8}$ in this work)

Require: Initial parameters ω

Initialize 1st and 2nd moment variables $\mathbf{s} = 0, \mathbf{r} = 0$

Initialize mini-batch counter $p = 0$

while stopping criterion not met **do**

 Sample a mini-batch of Q examples from the training set $\{\mathbf{x}_1, \dots, \mathbf{x}_Q\}$ with corresponding targets $\{\mathbf{y}_1, \dots, \mathbf{y}_Q\}$

 Compute gradient: $\mathbf{g} \leftarrow \nabla_{\omega} \sum_q \mathcal{L}(\mathcal{N}(\mathbf{x}_q; \omega), \mathbf{y}_q)$

$p \leftarrow p + 1$

 Update biased first moment estimate: $\mathbf{s} \leftarrow \rho_1 \mathbf{s} + (1 - \rho_1) \mathbf{g}$

 Update biased second moment estimate: $\mathbf{r} \leftarrow \rho_2 \mathbf{r} + (1 - \rho_2) \mathbf{g} \odot \mathbf{g}$

 Correct bias in first moment: $\hat{\mathbf{s}} \leftarrow \frac{\mathbf{s}}{1 - \rho_1^p}$

 Correct bias in second moment: $\hat{\mathbf{r}} \leftarrow \frac{\mathbf{r}}{1 - \rho_2^p}$

 Compute update: $\Delta\theta = -\epsilon \frac{\hat{\mathbf{s}}}{\sqrt{\hat{\mathbf{r}} + \delta}}$ (operations applied element-wise)

 Apply update: $\omega \leftarrow \omega + \Delta\omega$

end while

To use this optimizer, the training dataset is divided into smaller matrices $[\hat{\mathbf{x}}_q]_{q=1}^Q \in \mathbb{R}^{Q \times n_x}$ and $[\hat{\mathbf{y}}_q]_{q=1}^Q \in \mathbb{R}^{Q \times n_y}$, referred to as mini-batches, each containing $Q < N$ measurements. It is important to notice that the Q samples are selected randomly to prevent the MLP from learning any temporal correlation between consecutive observations.

It is also necessary to define a loss function that quantifies the discrepancy between the model predictions and the measurements in each mini-batch. In this work, the loss function is

$$\mathcal{L}(\omega) = \lambda_1 \sum_{q=1}^Q \mathcal{L}_q(\omega) = \lambda_1 \sum_{q=1}^Q \|\mathbf{y}_q - \hat{\mathbf{y}}_q\|_1,$$

where $\mathbf{y}_q = \mathcal{N}(\mathbf{x}_q; \omega)$ and $\lambda_1 = 10$ is a weighting factor. The 1-norm is employed instead of the more commonly used 2-norm because it yielded improved prediction accuracy in this project. This improvement is likely due to the 1-norm's gradient being constant (except at zero), whereas the gradient of the 2-norm diminishes as the prediction error becomes small.

It is important to note that the 1-norm is not differentiable when the prediction $\mathbf{y}_q$ exactly matches the measurement $\hat{\mathbf{y}}_q$. However, this is not a concern because subgradients can be used at such non-differentiable points, and convex functions like the 1-norm always admit subgradients [27].

Definition 2.1.1 (Subgradients) *Let $\mathcal{X} \subset \mathbb{R}^n$ and $f : \mathcal{X} \rightarrow \mathbb{R}$. Then $g \in \mathbb{R}^n$ is a subgradient of f at $x \in \mathcal{X}$ if for any $y \in \mathcal{X}$ one has*

$$f(y) \geq f(x) + g^T(y - x).$$

The gradient of the loss function with respect to the parameters of the MLP $\mathbf{g}$ is computed using automatic differentiation. This technique enables the computation of derivatives of functions defined by computer programs with accuracy up to machine precision [28]. In practice, automatic differentiation frameworks compute subgradients at non-differentiable points for convex functions [29], and no convergence issues were encountered in this work.

Finally, a separate validation dataset is used to define the stopping criterion. After each update of the parameters ω , the loss is evaluated on the validation set and tracked throughout training. The optimization process is terminated if the validation loss fails to decrease over a predefined number of iterations, which indicates that the model is overfitting: it has begun to memorize noise or fluctuations in the training data rather than learning generalizable patterns, resulting in degraded performance on unseen inputs.

2.2 Neural Ordinary Differential Equations

There are two primary approaches to estimate the differential equations that govern the system’s dynamics using neural networks: Residual Networks and Neural Ordinary Differential Equations (NODEs).

Residual Networks (ResNets) are a type of deep neural network architecture designed to ease the training of very deep models by introducing skip connections, or residual connections. These connections allow the input of a layer to bypass one or more layers and be added directly to the output. ResNets have been highly successful in image recognition tasks and were first introduced in [30]. For a time series problem, it can be formally expressed as

$$\mathbf{x}_{i+1} = \mathbf{x}_i + \mathcal{N}(\mathbf{x}_i, \omega)$$

where $\mathbf{x}_i$ represents the system state at time $t_i \in [0, T]$, $\mathcal{N}$ is a neural network that learns the difference between consecutive states and ω is its adjustable weights and biases. This formulation is mathematically equivalent to the explicit Euler integrator, which is the most basic explicit method.

Neural ODEs (NODEs) provide a more robust alternative by learning the vector field that governs the system’s evolution. Instead of modeling discrete state differences, they approximate $\dot{\mathbf{x}}$ using a neural network $\mathcal{N}$. Introduced in [31], NODEs allow for combining machine learning with differential equations.

The first step to applying them is to ensure that the initial value problem

$$\dot{\mathbf{x}} = \mathcal{N}(\mathbf{x}; \omega), \quad \mathbf{x}(t_0) = \mathbf{x}_0,$$

admits a well-defined solution. If the neural network $\mathcal{N}$ is constructed from compositions of Lipschitz continuous functions, then it is itself Lipschitz. Since the activation functions used in this project (ReLU and ELU) are Lipschitz continuous, $\mathcal{N}$ satisfies the conditions

of Picard’s existence and uniqueness theorem [32], thereby guaranteeing the existence and uniqueness of the solution to the initial value problem.

Theorem 2.2.1 (Picard’s Existence Theorem) *Let $f : [0, T] \times \mathbb{R}^d \rightarrow \mathbb{R}^d$ be continuous in t and Lipschitz continuous in $\mathbf{x}$. That is, there exists a constant $C > 0$ such that $\forall t \in [0, T]$ and all $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^d$,*

$$\|f(t, \mathbf{x}_1) - f(t, \mathbf{x}_2)\| \leq C\|\mathbf{x}_1 - \mathbf{x}_2\|.$$

Then, for any initial condition $\mathbf{x}_0 \in \mathbb{R}^d$, there exists a unique continuously differentiable function $\mathbf{x} : [0, T] \rightarrow \mathbb{R}^d$ satisfying the initial value problem

$$\dot{\mathbf{x}} = f(\mathbf{x}, t), \quad \mathbf{x}(t_0) = \mathbf{x}_0.$$

Therefore, starting from the initial state $\mathbf{x}_0$, we can define the state $\mathbf{x}_1 = \mathbf{x}(t_0 + \tau)$ of the system as

$$\mathbf{x}_1 = \mathbf{x}_0 + \int_{t_0}^{t_0 + \tau} \mathcal{N}(\mathbf{x}(t), \boldsymbol{\omega}) dt,$$

where the integral is approximated using a numerical integration scheme. This approach represents a significant improvement over Residual Networks since the study of numerical integration schemes is well-established and modern ODE solvers provide error control [31]. These solvers monitor approximation errors and dynamically adjust their evaluation strategy to maintain the desired accuracy while optimizing computational efficiency.

2.2.1 Training Process

Let f be an unknown function governing the time evolution of the state $\mathbf{x}(t)$:

$$\dot{\mathbf{x}} = f(\mathbf{x}(t)).$$

Assume that $N \in \mathbb{N}$ measurements of $\mathbf{x}(t)$ sampled at a fixed time-step τ are available. These are organized as a column-wise matrix $\hat{\mathbf{x}} = [\hat{\mathbf{x}}_j]_{j=0}^{N-1} \in \mathbb{R}^{n_x \times N}$. Each column $\hat{\mathbf{x}}_j = [\hat{\mathbf{x}}_j^{(i)}]_{i=0}^{n_x} \in \mathbb{R}^{n_x}$ contains the recorded state at time $t_j \in [0, T]$ with $T = N\tau$.

Let $\mathcal{N}(\mathbf{x}; \boldsymbol{\omega})$ denote a multilayer perceptron (MLP) parameterized by weights and biases $\boldsymbol{\omega}$. Then, the Neural Ordinary Differential Equation (NODE) replaces the unknown dynamics f with the MLP:

$$\dot{\mathbf{x}} = \mathcal{N}(\mathbf{x}(t); \boldsymbol{\omega}).$$

Training the NODE also aims to find the optimal parameters $\boldsymbol{\omega}^*$ that minimize a loss function $\mathcal{L}(\boldsymbol{\omega})$. In this project, this optimization is also performed using the ADAM optimizer.

Again, the training dataset is divided into the mini-batches $[\hat{\mathbf{x}}_q]_{q=1}^Q \in \mathbb{R}^{Q \times n_x}$. However, in contrast to the training process described in Subsection 2.1.1, each mini-batch corresponds to a trajectory sampled from a time t_0 to a final time $t_f = t_0 + \tau Q$ since the aim is to learn the temporal evolution of $\mathbf{x}(t)$. For a given batch, predictions are generated by initializing the model at $\mathbf{x}_0 = \hat{\mathbf{x}}_0$ and integrating the NODE to produce $\mathbf{x}_1$, then using $\mathbf{x}_1$ to compute $\mathbf{x}_2$, and so on, until the final time step t_f .

In this project, the loss function that quantifies the difference between the predicted and measured trajectories within each mini-batch is defined as

$$\mathcal{L}(\omega) = \lambda_2 \|\mathbf{x}_1 - \hat{\mathbf{x}}_1\|_1 + \lambda_3 \sum_{q=1}^Q \|\mathbf{x}_q - \hat{\mathbf{x}}_q\|_1.$$

where $\lambda_2 = 1$ and $\lambda_3 = 5$ are weighting factors. The first predicted state $\mathbf{x}_1$ is considered twice in the loss function to assign it a higher weight, as it establishes the starting point for the trajectory and strongly influences the accuracy of subsequent predictions. Errors in the initial step often propagate and amplify over time; therefore, assigning them a higher penalty encourages the model to follow the true system dynamics more closely from the beginning.

The 1-norm is again preferred over the 2-norm as it yielded better prediction accuracy. Additionally, in preliminary experiments it was observed that training with the 2-norm often failed to converge when residuals were large, particularly in the early stages. This behavior was likely caused by excessively large gradients, which prevented the optimizer from reducing the loss effectively. In contrast, the 1-norm penalizes errors linearly, resulting in more consistent gradient magnitudes and reducing the risk of gradient explosions that hinder convergence.

There are three main approaches for computing the gradients of the loss with respect to the model parameters when using NODEs [33].

- BackPropagation Through Time (BPTT). This is the most widely used method due to its ease of implementation and fast execution. However, it is memory-intensive, as all intermediate states and solver operations must be stored during the forward pass for use in the backward pass.
- Adjoint method. Gradients are computed by solving a second, augmented ODE backward in time. This method is more memory-efficient since the intermediate steps are recomputed during the backward pass, not requiring to store them. However, this recomputation increases computational cost and may introduce numerical errors, resulting in less accurate gradients than those obtained using BPTT [34].
- Reversible ODEs. This method provides a good balance between computational efficiency and gradient accuracy [35]. Nevertheless, it is not supported by the software framework used in this thesis and is therefore not considered further.

We employ Backpropagation Through Time (BPTT) because memory is not a limiting factor for the models developed in this work and provides more accurate gradients than the adjoint method. To illustrate the computations, we consider the loss function

$$\mathcal{L}(\omega) = \sum_{q=1}^Q \mathcal{L}_q(\omega) = \sum_{q=1}^Q \|\mathbf{x}_q - \hat{\mathbf{x}}_q\|_1,$$

where $\mathbf{x}_q$ represents the model prediction at time $t_q = t_0 + \tau q$, for $q = 1, \dots, Q$. For simplicity, we will assume that the integral of the NODE is computed using the Explicit Euler method. Hence,

$$\mathbf{x}_q = \mathbf{x}_{q-1} + \tau \mathcal{N}(\mathbf{x}_{q-1}; \omega). \quad (2.2.1)$$

The gradient of the loss function with respect to a particular parameter $\omega_{ab}^{(c)}$ is

$$\frac{\partial \mathcal{L}}{\partial \omega_{ab}^{(c)}} = \frac{\partial \mathcal{L}}{\partial \mathbf{x}_i} \frac{\partial \mathbf{x}_i}{\partial \omega_{ab}^{(c)}} = \sum_{q=1}^Q \frac{\partial \mathcal{L}_q}{\partial \mathbf{x}_i} \frac{\partial \mathbf{x}_i}{\partial \omega_{ab}^{(c)}}.$$

First,

$$\sum_{q=1}^Q \frac{\partial \mathcal{L}_q}{\partial \mathbf{x}_i} = \frac{\partial \mathcal{L}_i}{\partial \mathbf{x}_i} + \frac{\partial \mathcal{L}_{i+1}}{\partial \mathbf{x}_i} = \frac{\partial \mathcal{L}_i}{\partial \mathbf{x}_i} + \frac{\partial \mathbf{x}_{i+1}}{\partial \mathbf{x}_i} \frac{\partial \mathcal{L}_{i+1}}{\partial \mathbf{x}_{i+1}} = \frac{\partial \mathcal{L}_i}{\partial \mathbf{x}_i} + \left(\mathbf{I} + \tau \frac{\partial \mathcal{N}(\mathbf{x}_i; \boldsymbol{\omega})}{\partial \mathbf{x}_i} \right) \frac{\partial \mathcal{L}_{i+1}}{\partial \mathbf{x}_{i+1}},$$

where in the first equality it was used that the prediction $\mathbf{x}_q$ depends only on its immediate predecessor, according to the recursive structure of the Explicit Euler scheme.

From (2.2.1), the derivative of the state with respect to the parameter $\omega_{ab}^{(c)}$ satisfies

$$\begin{aligned} \frac{\partial \mathbf{x}_{i+1}}{\partial \omega_{ab}^{(c)}} &= \frac{\partial \mathbf{x}_i}{\partial \omega_{ab}^{(c)}} + \tau \frac{d\mathcal{N}(\mathbf{x}_i; \boldsymbol{\omega})}{d\omega_{ab}^{(c)}} \\ &= \frac{\partial \mathbf{x}_i}{\partial \omega_{ab}^{(c)}} + \tau \left(\frac{\partial \mathcal{N}(\mathbf{x}_i; \boldsymbol{\omega})}{\partial \omega_{ab}^{(c)}} + \frac{\partial \mathcal{N}(\mathbf{x}_i; \boldsymbol{\omega})}{\partial \mathbf{x}_i} \frac{\partial \mathbf{x}_i}{\partial \omega_{ab}^{(c)}} \right) \\ &= \tau \frac{\partial \mathcal{N}(\mathbf{x}_i; \boldsymbol{\omega})}{\partial \omega_{ab}^{(c)}} + \left(\mathbf{I} + \tau \frac{\partial \mathcal{N}(\mathbf{x}_i; \boldsymbol{\omega})}{\partial \mathbf{x}_i} \right) \frac{\partial \mathbf{x}_i}{\partial \omega_{ab}^{(c)}}, \end{aligned}$$

with $\frac{\partial \mathbf{x}_0}{\partial \omega_{ab}^{(c)}} = 0$ since $\mathbf{x}_0 = \hat{\mathbf{x}}_0$.

This is a linear recurrence of the form

$$\mathbf{a}_i = \mathbf{b}_i + \underline{\mathbf{c}}_i \mathbf{a}_{i-1}$$

where

$$\mathbf{a}_i = \frac{\partial \mathbf{x}_i}{\partial \omega_{ab}^{(c)}}, \quad \mathbf{b}_i = \tau \frac{\partial \mathcal{N}(\mathbf{x}_{i-1}; \boldsymbol{\omega})}{\partial \omega_{ab}^{(c)}} \quad \text{and} \quad \underline{\mathbf{c}}_i = \mathbf{I} + \tau \frac{\partial \mathcal{N}(\mathbf{x}_{i-1}; \boldsymbol{\omega})}{\partial \mathbf{x}_{i-1}}.$$

By recursively substituting $\mathbf{a}_{i-1}, \mathbf{a}_{i-2}, \dots$, we obtain the closed-form expression:

$$\mathbf{a}_i = \mathbf{b}_i + \sum_{j=1}^{i-1} \left(\prod_{k=j+1}^i \underline{\mathbf{c}}_k \right) \mathbf{b}_j.$$

Therefore,

$$\frac{\partial \mathbf{x}_i}{\partial \omega_{ab}^{(c)}} = \tau \frac{\partial \mathcal{N}(\mathbf{x}_i; \boldsymbol{\omega})}{\partial \omega_{ab}^{(c)}} + \sum_{j=1}^{i-1} \left(\prod_{k=j+1}^i \left[\mathbf{I} + \tau \frac{\partial \mathcal{N}(\mathbf{x}_{k-1}; \boldsymbol{\omega})}{\partial \mathbf{x}_{k-1}} \right] \right) \tau \frac{\partial \mathcal{N}(\mathbf{x}_j; \boldsymbol{\omega})}{\partial \omega_{ab}^{(c)}},$$

where the terms

$$\frac{\partial \mathcal{N}(\mathbf{x}_i; \boldsymbol{\omega})}{\partial \omega_{ab}^{(c)}} \quad \text{and} \quad \frac{\partial \mathcal{N}(\mathbf{x}_i; \boldsymbol{\omega})}{\partial \mathbf{x}_i}$$

are computed via automatic differentiation.

The stopping criterion is also defined using a validation dataset. The loss is evaluated on this dataset every time the parameters $\boldsymbol{\omega}$ are updated. The results are tracked throughout the training process, which finishes if the validation loss does not decrease after a predefined number of iterations.

Chapter 3

System Description and Modeling Framework

3.1 Modeling Tools

This thesis employs two primary tools for modeling and simulation: Dymola and NeuroMANCER.

Dymola (Dynamic Modeling Laboratory) is a commercial modeling and simulation environment based on the Modelica language. It is specifically designed for the modeling of complex physical systems involving multiple domains such as thermodynamics, fluid dynamics, and control [36]. In this thesis, Dymola is utilized to simulate the chiller plant and to generate the training, validation, and test datasets used in this work.

NeuroMANCER (Neural Modules with Adaptive Nonlinear Constraints and Efficient Regularizations) is an open-source differentiable programming library developed by the Pacific Northwest National Laboratory (PNNL). Built on top of PyTorch, NeuroMANCER is designed for solving parametric constrained optimization problems, physics-informed system identification, and parametric model predictive control. It facilitates the integration of machine learning with scientific computing by enabling the construction of end-to-end differentiable models embedded with prior knowledge and physical constraints [37]. In this work, version 1.5.2 of NeuroMANCER is employed. A key feature of the library is its support for Neural Ordinary Differential Equations (NODEs), which are used in this thesis to construct both gray-box and black-box models of the chiller plant.

3.2 System of Study

The Modelica model of the chiller plant enables access to internal variables that are typically unmeasurable in real-world systems, such as the heat exchanged in the chiller's evaporator. This limitation arises because, in real systems, measurements are obtained using temperature sensors and flow meters, which are also used to control the plant operation via predefined setpoints. Additionally, power sensors are used to measure the power consumption of each component in the plant. As a result, only a subset of the variables available in the Modelica model can be leveraged for system identification.

3. SYSTEM DESCRIPTION AND MODELING FRAMEWORK

For the system under study, temperature sensors are positioned after each component that alters the water temperature, while flow meters are placed downstream of the pumps, as illustrated in [Figure 3.1](#). These sensors are connected to the Building Management System (BMS), which adjusts the operation of the individual components to ensure that each measured variable reaches its corresponding setpoint. Therefore, setpoints serve as control inputs and define the target temperatures of the chilled and cooling water at specific points in the system, ensuring a safe temperature within the IT room.

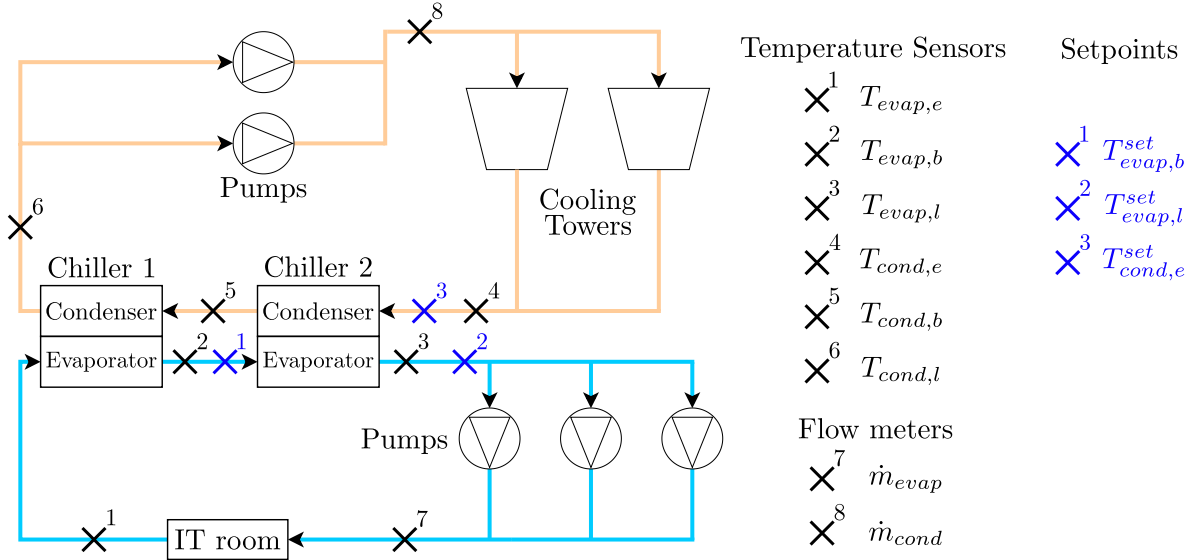


Figure 3.1: Schematic of the chiller plant that will be modeled. The system comprises two chillers, pumps, cooling towers, and an IT room as the cooling load. Temperature sensors and flow meters are placed after each component to monitor key variables, which are used to control the system through temperature setpoints. A description of all the variables that will be used to model the chiller plant can be found in [Table 3.1](#)

This work will focus on three setpoints: $T_{evap,l}^{set}$, $T_{evap,b}^{set}$ and $T_{cond,e}^{set}$. These correspond to the desired temperature of the chilled water after the second and first chiller, and the desired temperature of the cooling water after the cooling towers, respectively.

In addition, the Modelica model provides access to two key performance indicators of the chillers: the Part Load Ratio (PLR) and the Coefficient of Performance (COP). While these variables cannot be directly measured in real systems, they can be estimated using measurable quantities such as the temperature and mass flow rate of the water entering and exiting each chiller. Modeling these variables is essential because they provide insight into the operating conditions and efficiency of the system, which in turn influences control strategies, energy consumption forecasts, and overall system optimization.

External weather conditions must also be taken into account, as they influence the performance of the cooling towers. For this reason, real-time measurements of the dry-bulb temperature and relative humidity are collected. Together with the IT load, which generates the heat absorbed by the chilled water, these factors are incorporated into the model as disturbances, since they cannot be controlled.

Variable	Description	Units
$T_{evap,e}$	Temperature of the chilled water entering in chiller 1.	°C
$T_{evap,b}$	Temperature of the chilled water leaving chiller 1 and entering chiller 2.	°C
$T_{evap,l}$	Temperature of the downstream chilled water.	°C
$T_{cond,e}$	Temperature of the cooling water entering in chiller 2.	°C
$T_{cond,b}$	Temperature of the chilled water leaving chiller 2 and entering chiller 1.	°C
$T_{cond,l}$	Temperature of the cooling water leaving chiller 2.	°C
$T_{evap,b}^{set}$	Setpoint for $T_{evap,b}$.	°C
$T_{evap,l}^{set}$	Setpoint for $T_{evap,l}$.	°C
$T_{cond,e}^{set}$	Setpoint for $T_{cond,e}$.	°C
$\dot{m}_{evap}$	Mass flow rate in the evaporator circuit.	kg/s
P_1	Power consumption of chiller 1.	kW
P_2	Power consumption of chiller 2.	kW
P_{ct}	Power consumption of the cooling tower.	kW
P_{ahu}	Power consumption of the air handling unit in the IT room.	kW
$P_{p,evap}$	Power consumption of the evaporator pump.	kW
COP_1	Coefficient of performance of chiller 1.	–
COP_2	Coefficient of performance of chiller 2.	–
PLR_1	Part-load ratio of chiller 1.	–
PLR_2	Part-load ratio of chiller 2.	–
T_{db}	Dry-bulb temperature.	°C
RH	Relative humidity.	%
$\dot{Q}_{load}$	Cooling load of the IT room.	MW

Table 3.1: Description of the variables that will be used to model the chiller plant.

Lastly, the mass flow rate in the condenser circuit ($\dot{m}_{cond}$) is assumed to be constant since it remains approximately steady in practice. Consequently, together with the power consumption of the condenser pump ($P_{p,cond}$), these variables are excluded from the model, as they do not vary. All the variables that will be used to model the chiller plant are collected in [Table 3.1](#).

3.3 System Modeling Equations

This section presents the theory-based structure used to construct the gray-box model of the chiller plant. The model comprises first-order differential equations that describe the evolution of the temperature variables, as well as algebraic expressions used to estimate the chiller outputs.

3.3.1 Equations for the Evolution of the Temperature Variables

The Modelica model (ground truth) is employed to analyze the dynamics of each component within the chiller plant, with the aim of deriving the differential equations that will

form the basis of the gray-box models. To accomplish this, a step response experiment is conducted for each component. As an illustrative example, we focus on the evaporator of the second chiller.

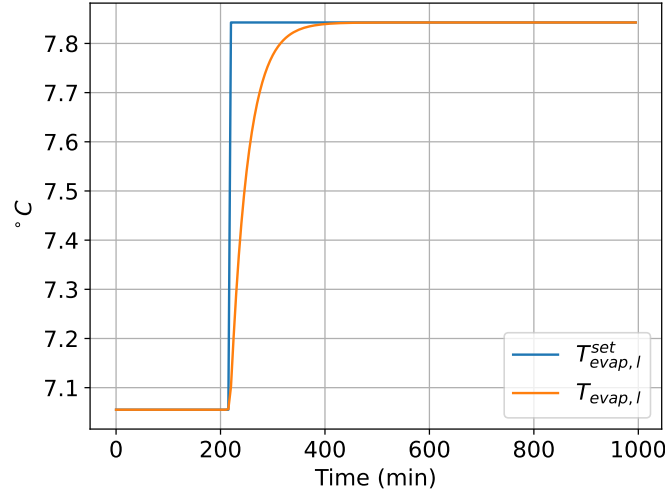


Figure 3.2: Step response of the temperature of the water leaving the evaporator (orange) following a step change in its setpoint (blue).

In this case, the temperature of the chilled water leaving the evaporator $T_{evap,l}$ is expected to be influenced by the temperature of the water entering the evaporator $T_{evap,b}$ and the setpoint $T_{evap,l}^{set}$ due to the Building Management System (BMS) action. To investigate this relationship, the system is first allowed to reach a steady state. Once steady conditions are achieved, the $T_{evap,l}^{set}$ (or $T_{evap,b}$) is perturbed using a step function as illustrated in Figure 3.2, and the response of the outlet chilled water temperature is observed and analyzed.

To do this, we consider the first-order differential equation

$$\dot{T}_{evap,l} = -\frac{1}{\kappa}T_{evap,l} + \frac{K}{\kappa}\Delta T_{evap,l}^{set}$$

where $\Delta T_{evap,l}^{set}$ is the amplitude of the step function that was used to excite the setpoint. The time constant of the system τ represents the time it takes for $T_{evap,l}$ to change from its steady state value $T_{evap,l}^{steady}$ to $T_{evap,l}^{steady} + (1 - e^{-1})K\Delta T_{evap,l}^{set}$, with

$$K = \frac{\Delta T_{evap,l}}{\Delta T_{evap,l}^{set}} \quad (3.3.1)$$

being a gain parameter that determines the ratio between the amplitude of the step response $\Delta T_{evap,l}$ and $\Delta T_{evap,l}^{set}$. With this, it is possible to compute the value of κ for this experiment.

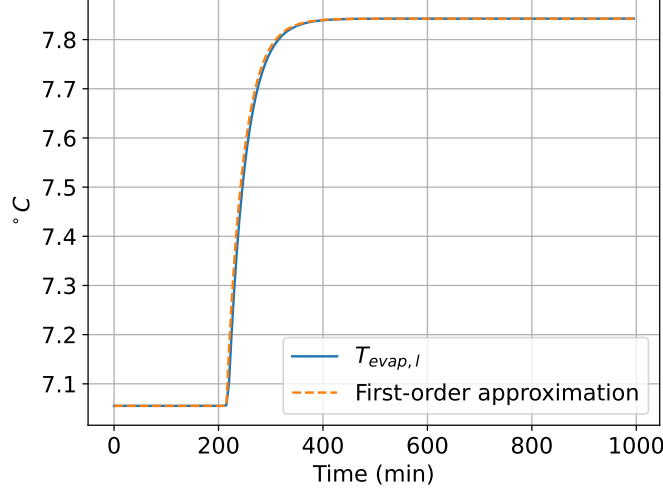


Figure 3.3: Comparison between the step response (solid blue line) of $T_{evap,l}$ and its first-order approximation (dashed orange line).

As it can be observed in Figure 3.3, the first-order approximation follows closely the step response observed using the Modelica model for $T_{evap,l}$. This experiment can be applied by exciting $T_{evap,b}$ with similar results. Therefore, we define the differential equation for the evaporator of the second chiller:

$$\dot{T}_{evap,l} = c_0 T_{evap,l}^{set} - c_1 T_{evap,l} + c_2 T_{evap,b} + c_3$$

where $c_i \in \mathbb{R}$ for $i = 0, \dots, 3$.

This procedure is repeated for the rest of the components of the chiller plant to obtain the rest of the first-order differential equations:

$$\begin{aligned} \dot{T}_{evap,e} &= a_0 \dot{Q}_{load} - a_1 T_{evap,e} + a_2 T_{evap,l} + a_3, \\ \dot{T}_{evap,b} &= b_0 T_{evap,b}^{set} - b_1 T_{evap,b} + b_2 T_{evap,e} + b_3, \\ \dot{T}_{cond,e} &= d_0 T_{cond,e}^{set} - d_1 T_{cond,e} + d_2 T_{cond,l} + d_3, \\ \dot{T}_{cond,b} &= -e_0 T_{evap,l}^{set} - e_1 T_{cond,b} + e_2 T_{cond,e} + e_3, \\ \dot{T}_{cond,l} &= -f_0 T_{evap,b}^{set} - f_1 T_{cond,l} + f_2 T_{cond,b} + f_3. \end{aligned}$$

In this project, the parameters $a_i, b_i, c_i, d_i, e_i, f_i \in \mathbb{R}$ for $i = 0, \dots, 3$ will be estimated using NeuroMANCER.

3.3.2 Equations for the Outputs of a Chiller

The part load ratio quantifies the ratio between the cooling load at the evaporator and the chiller's maximum cooling capacity. It is a dimensionless quantity typically ranging from 0 (chiller is off) to 1 (full load). The PLR is given by

$$PLR = \frac{\dot{Q}_{evap}}{\alpha},$$

where $\dot{Q}_{evap}$ is the heat exchange in the evaporator of the chiller and $\alpha = \dot{Q}_{max} > 0$ is the maximum cooling capacity of the chiller specified by the manufacturer.

For instance, the heat exchange in the evaporator of the first chiller is given by

$$\dot{Q}_{evap} = \dot{m}_{evap} c_p (T_{evap,e} - T_{evap,b}),$$

where c_p is the water specific heat capacity.

To model the power consumption (P), we employ an expression obtained using the ALAMO (Automatic Learning of Algebraic MOdels) framework. ALAMO constructs low-complexity, physically interpretable models from data by selecting a sparse set of nonlinear basis functions that best capture the behavior of the target variable [38]. In this case, it yields a compact representation of the chiller's power consumption as a function of relevant operating variables. For the first chiller, the expression reads:

$$\begin{aligned} & \text{ALAMO}(\beta, T_{cond,b}, T_{evap,b}, PLR_1) \\ &= 1000 \left(\beta_0 + (\beta_1 T_{cond,b} PLR_1 - \beta_2 T_{evap,b} + \beta_4) PLR_1 + \beta_3 T_{cond,b}^3 \right), \end{aligned} \quad (3.3.2)$$

where β_i for $i = \{0, 1, 2, 3, 4\}$ are model parameters specific of each chiller. An analogous expression applies to the second chiller by replacing $T_{cond,b}$, $T_{evap,b}$ and PLR_1 with $T_{cond,e}$, $T_{evap,l}$ and PLR_2 , respectively.

The coefficient of performance, which is a measure of its energy efficiency, is defined as the ratio of the useful cooling provided to the electrical power consumed:

$$COP = \frac{\dot{Q}_{evap}}{P}. \quad (3.3.3)$$

A higher COP indicates a more efficient chiller, as it means more cooling is achieved per unit of electrical energy consumed.

Finally, the value of $\dot{Q}_{max}$ and the parameters of the ALAMO model will be estimated from the training dataset using a least-squares approach.

Chapter 4

Methodology

4.1 Dataset Generation

As previously mentioned, the training, validation and test datasets are generated using the Modelica model. For this, it is necessary to define the disturbances $\mathbf{d}$ and control inputs $\mathbf{u}$ for each dataset as they are inputs of the Modelica model.

To do this, real weather data from a Florida weather station during Summer 2024 was collected [39]. The data contained hourly samples of the dry-bulb temperature T_{db} and the relative humidity RH . To obtain values between samples, linear interpolation was used.

Additionally, three different profiles for the IT load were employed:

- *Constant load*: Fixed at either 1.2 MW or 1.4 MW.
- *Sinusoidal load*: The periodicity is 4 hours with values oscillating in the range $[1.0, 1.8]$ MW.
- *Random step load*: The height of the step function was randomly sampled from a uniform distribution over $[1.0, 1.8]$ MW.

The profile for each setpoint also follows a random step function, where the height is updated hourly. The chilled water temperature setpoint, $T_{evap,l}^{set}$, was sampled from $[6.5, 8]^\circ\text{C}$, and the temperature difference setpoint, $T_{evap,\Delta}^{set}$, from $[2, 4]^\circ\text{C}$. Then, the setpoint $T_{evap,b}^{set} = T_{evap,l}^{set} + T_{evap,\Delta}^{set}$. The cooling tower setpoint, $T_{cond,e}^{set}$, was defined as the outdoor temperature plus a random offset sampled from $[-1, 2]^\circ\text{C}$.

There is a particular case where the setpoints are derived from the baseline rule-based closed-loop controller. In this case, only $T_{evap,\Delta}^{set}$ was varied, while the other setpoints were fixed at $T_{evap,l}^{set} = 6.67^\circ\text{C}$ and $T_{cond,e}^{set} = 29.4^\circ\text{C}$.

4.2 Neural Modeling

Let the system of interest be described by

$$\begin{cases} \dot{\mathbf{x}} = f_x(\mathbf{x}, \mathbf{u}, \mathbf{d}), \\ \mathbf{y} = f_y(\mathbf{x}, \mathbf{u}, \mathbf{d}), \end{cases}$$

where $\mathbf{x} \in \mathbb{R}^{n_x}$ are the system states, $\mathbf{y} \in \mathbb{R}^{n_y}$ the outputs, $\mathbf{u} \in \mathbb{R}^{n_u}$ the control inputs and $\mathbf{d} \in \mathbb{R}^{n_d}$ the disturbances. The function f_x governs the dynamics of the system state, and f_y connects it to the system's outputs.

Modeling this system analytically is usually difficult to achieve as the functions f_x and f_y may contain complex interactions and nonlinearities. Therefore, black-box and gray-box models are employed to estimate these functions from measurements of $\mathbf{x}$, $\mathbf{y}$, $\mathbf{u}$, $\mathbf{d}$.

In this project, black-box models are constructed using Multi-Layer Perceptrons (MLPs) to approximate the functions f_x and f_y . The resulting model is expressed as,

$$\begin{cases} \dot{\mathbf{x}} = \mathcal{N}_x(\mathbf{x}, \mathbf{u}, \mathbf{d}; \boldsymbol{\omega}_x), \\ \mathbf{y} = \mathcal{N}_y(\mathbf{x}, \mathbf{u}, \mathbf{d}; \boldsymbol{\omega}_y), \end{cases} \quad (4.2.1)$$

where $\mathcal{N}_x(\mathbf{x}, \mathbf{u}, \mathbf{d}; \boldsymbol{\omega}_x)$ and $\mathcal{N}_y(\mathbf{x}, \mathbf{u}, \mathbf{d}; \boldsymbol{\omega}_y)$ are MLPs with parameters $\boldsymbol{\omega}_x$ and $\boldsymbol{\omega}_y$, respectively.

In contrast, gray-box models combine theory-based functions with data-driven techniques. In this project, the general structure for gray-box models is given by

$$\begin{cases} \dot{\mathbf{x}} = \tilde{f}_x(\mathbf{x}, \mathbf{u}, \mathbf{d}) + \mathcal{N}_x(\mathbf{x}, \mathbf{u}, \mathbf{d}; \boldsymbol{\omega}_x), \\ \mathbf{y} = \tilde{f}_y(\mathbf{x}, \mathbf{u}, \mathbf{d}) + \mathcal{N}_y(\mathbf{x}, \mathbf{u}, \mathbf{d}; \boldsymbol{\omega}_y), \end{cases} \quad (4.2.2)$$

where $\tilde{f}_x$ and $\tilde{f}_y$ are approximations of the unknown system functions f_x and f_y , respectively. The function $\tilde{f}_x$ is derived by analyzing the system dynamics, as described in [Subsection 3.3.1](#), while $\tilde{f}_y$ is constructed using empirical models and definitions, as outlined in [Subsection 3.3.2](#). The terms $\mathcal{N}_x$ and $\mathcal{N}_y$ are MLPs that learn the residual dynamics not captured by the analytical approximations, thereby increasing the model's flexibility and accuracy. One example of such residual dynamics is the effect of flow rates on the evolution of water temperature, which is not captured by the first-order differential equations derived in [Subsection 3.3.1](#).

Each model structure can be further divided into non-component-based and component-based approaches. The black and gray-box non-component-based approaches are given by (4.2.1) and (4.2.2), respectively. In these configurations, a single MLP is used to model the system states $\mathbf{x}$ and another to predict the outputs $\mathbf{y}$.

In the component-based approach, each subsystem (each chiller, the cooling towers, the pumps and the combination of the IT room and the Air Handling Unit) is assigned a dedicated pair of neural networks: one network models the dynamics, and the other predicts the outputs.

This project first focuses on modeling a simplified system consisting of two isolated chillers, and subsequently, the complete chiller plant. The following subsections present a

detailed description of the variables used to model each system, along with the equations corresponding to the four modeling approaches.

4.2.1 Models for the Two-Chiller System

The isolated two-chiller system illustrated in Figure 4.1 will be modeled using Neural Ordinary Differential Equations (NODEs) to evaluate their suitability for representing dynamical systems. To this end, it is first necessary to identify the relevant variables and classify them into states, inputs, disturbances and outputs.

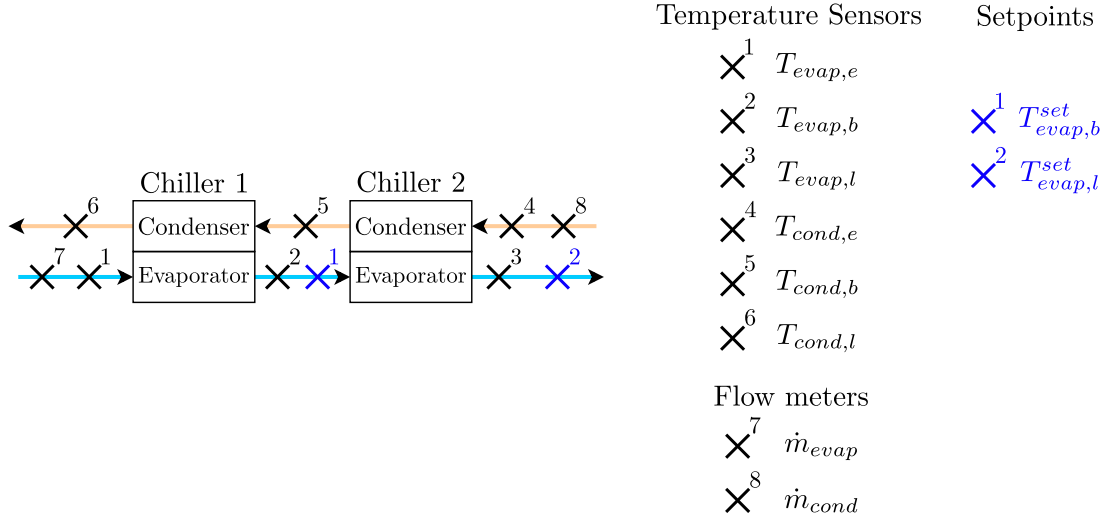


Figure 4.1: Schematic of the system of two chillers that will be modeled. Temperature sensors and flow meters are used to control the system through temperature setpoints.

Given that the system consists exclusively of chillers, which only modify the temperature of both condenser and chilled water, the mass flow of both circuits will not be modeled as states. Furthermore, the cooling-water mass flow rate is assumed constant and is therefore omitted from the model. External factors such as weather conditions or the cooling load of the IT room do not directly influence the system and will therefore be excluded from the model. Under these assumptions, the system state $\mathbf{x}$, control inputs $\mathbf{u}$, disturbances $\mathbf{d}$ and outputs $\mathbf{y}$ are defined as follows:

$$\mathbf{x} = [T_{evap,b}, T_{evap,l}, T_{cond,b}, T_{cond,l}]^T, \quad \mathbf{u} = [T_{evap,b}^{set}, T_{evap,l}^{set}]^T, \\ \mathbf{d} = [\dot{m}_{evap}, T_{evap,e}, T_{cond,e}]^T, \quad \mathbf{y} = [PLR_1, P_1, COP_1, PLR_2, P_2, COP_2]^T.$$

Once the variables have been chosen, it is possible to present the equations that will be used in each modeling approach. It is important to notice that $\mathcal{N}$ represents a Multi-Layer Perceptron (MLP) with parameters ω .

- Black-box non-component-based:

$$\begin{cases} \dot{\mathbf{x}} = \mathcal{N}_1(\boldsymbol{\xi}; \omega_1), \\ \mathbf{y} = \mathcal{N}_2(\boldsymbol{\xi}; \omega_2), \end{cases}$$

where $\xi = \begin{bmatrix} \mathbf{x} \\ \mathbf{u} \\ \mathbf{d} \end{bmatrix}$.

- Black-box component-based:

$$\dot{\mathbf{x}} = \begin{pmatrix} \dot{T}_{evap,b} \\ \dot{T}_{evap,l} \\ \dot{T}_{cond,b} \\ \dot{T}_{cond,l} \end{pmatrix} = \begin{pmatrix} \mathcal{N}_1^{(1)}(\xi_{ch1}; \omega_1) \\ \mathcal{N}_2^{(1)}(\xi_{ch2}; \omega_2) \\ \mathcal{N}_1^{(2)}(\xi_{ch1}; \omega_1) \\ \mathcal{N}_2^{(2)}(\xi_{ch2}; \omega_2) \end{pmatrix}, \quad \mathbf{y} = \begin{pmatrix} PLR_1 \\ P_1 \\ COP_1 \\ PLR_2 \\ P_2 \\ COP_2 \end{pmatrix} = \begin{pmatrix} \mathcal{N}_3^{(1)}(\xi_{ch1}; \omega_3) \\ \mathcal{N}_3^{(2)}(\xi_{ch1}; \omega_3) \\ \mathcal{N}_3^{(3)}(\xi_{ch1}; \omega_3) \\ \mathcal{N}_4^{(1)}(\xi_{ch2}; \omega_4) \\ \mathcal{N}_4^{(2)}(\xi_{ch2}; \omega_4) \\ \mathcal{N}_4^{(3)}(\xi_{ch2}; \omega_4) \end{pmatrix}.$$

where

$$\xi_{ch1} = [T_{cond,b}, T_{cond,l}, T_{evap,e}, T_{evap,b}, T_{evap,b}^{set}, \dot{m}_{evap}]^T$$

and

$$\xi_{ch2} = [T_{cond,e}, T_{cond,b}, T_{evap,b}, T_{evap,l}, T_{evap,l}^{set}, \dot{m}_{evap}]^T.$$

Here, $\mathcal{N}_i^{(j)}$ denotes the j th node of the output layer of the i th MLP. This notation will be used consistently throughout the remainder of the section.

- Gray-box non-component-based:

$$\dot{\mathbf{x}} = \begin{pmatrix} \dot{T}_{evap,b} \\ \dot{T}_{evap,l} \\ \dot{T}_{cond,b} \\ \dot{T}_{cond,l} \end{pmatrix} = \begin{pmatrix} b_0 T_{evap,b}^{set} - b_1 T_{evap,b} + b_2 T_{evap,e} + b_3 + \mathcal{N}_1^{(1)}(\xi; \omega_1) \\ c_0 T_{evap,l}^{set} - c_1 T_{evap,l} + c_2 T_{evap,b} + c_3 + \mathcal{N}_1^{(2)}(\xi; \omega_1) \\ -e_0 T_{evap,l}^{set} - e_1 T_{cond,b} + e_2 T_{cond,e} + e_3 + \mathcal{N}_1^{(3)}(\xi; \omega_1) \\ -f_0 T_{evap,b}^{set} - f_1 T_{cond,l} + f_2 T_{cond,b} + f_3 + \mathcal{N}_1^{(4)}(\xi; \omega_1) \end{pmatrix},$$

$$\mathbf{y} = \begin{pmatrix} PLR_1 \\ P_1 \\ COP_1 \\ PLR_2 \\ P_2 \\ COP_2 \end{pmatrix} = \begin{pmatrix} \frac{\dot{Q}_{evap}}{\alpha_1} + \mathcal{N}_2^{(1)}(\xi; \omega_2) \\ \text{ALAMO}(\beta_1, T_{cond,b}, T_{evap,b}, PLR_1) + \mathcal{N}_2^{(2)}(\xi; \omega_2) \\ \frac{\dot{Q}_{evap}}{P_1} + \mathcal{N}_2^{(3)}(\xi; \omega_2) \\ \frac{\dot{Q}_{evap}}{\alpha_2} + \mathcal{N}_2^{(4)}(\xi; \omega_2) \\ \text{ALAMO}(\beta_2, T_{cond,e}, T_{evap,l}, PLR_2) + \mathcal{N}_2^{(5)}(\xi; \omega_2) \\ \frac{\dot{Q}_{evap}}{P_2} + \mathcal{N}_2^{(6)}(\xi; \omega_2) \end{pmatrix}.$$

- Gray-box component-based:

$$\dot{\mathbf{x}} = \begin{pmatrix} \dot{T}_{evap,b} \\ \dot{T}_{evap,l} \\ \dot{T}_{cond,b} \\ \dot{T}_{cond,l} \end{pmatrix} = \begin{pmatrix} b_0 T_{evap,b}^{set} - b_1 T_{evap,b} + b_2 T_{evap,e} + b_3 + \mathcal{N}_1^{(1)}(\xi_{ch1}; \omega_1) \\ c_0 T_{evap,l}^{set} - c_1 T_{evap,l} + c_2 T_{evap,b} + c_3 + \mathcal{N}_2^{(1)}(\xi_{ch2}; \omega_2) \\ -e_0 T_{evap,l}^{set} - e_1 T_{cond,b} + e_2 T_{cond,e} + e_3 + \mathcal{N}_1^{(2)}(\xi_{ch1}; \omega_1) \\ -f_0 T_{evap,b}^{set} - f_1 T_{cond,l} + f_2 T_{cond,b} + f_3 + \mathcal{N}_2^{(2)}(\xi_{ch2}; \omega_2) \end{pmatrix},$$

$$\mathbf{y} = \begin{pmatrix} PLR_1 \\ P_1 \\ COP_1 \\ PLR_2 \\ P_2 \\ COP_2 \end{pmatrix} = \begin{pmatrix} \frac{\dot{Q}_{evap}}{\alpha_1} + \mathcal{N}_3^{(1)}(\boldsymbol{\xi}_{ch1}; \boldsymbol{\omega}_3) \\ \text{ALAMO}(\boldsymbol{\beta}_1, T_{cond,b}, T_{evap,b}, PLR_1) + \mathcal{N}_3^{(2)}(\boldsymbol{\xi}_{ch1}; \boldsymbol{\omega}_3) \\ \frac{\dot{Q}_{evap}}{P_1} + \mathcal{N}_3^{(3)}(\boldsymbol{\xi}_{ch1}; \boldsymbol{\omega}_3) \\ \frac{\dot{Q}_{evap}}{\alpha_2} + \mathcal{N}_4^{(1)}(\boldsymbol{\xi}_{ch2}; \boldsymbol{\omega}_4) \\ \text{ALAMO}(\boldsymbol{\beta}_2, T_{cond,e}, T_{evap,l}, PLR_2) + \mathcal{N}_4^{(2)}(\boldsymbol{\xi}_{ch2}; \boldsymbol{\omega}_4) \\ \frac{\dot{Q}_{evap}}{P_2} + \mathcal{N}_4^{(3)}(\boldsymbol{\xi}_{ch2}; \boldsymbol{\omega}_4) \end{pmatrix}.$$

4.2.2 Models for the Complete Chiller Plant

The different models of the plant will be used to perform several experiments that will illustrate the effects of varying the number of parameters and the amount of training data, among others. For this, it is first necessary to classify the variables collected in [Table 3.1](#) into states $\mathbf{x}$, inputs $\mathbf{u}$, disturbances $\mathbf{d}$ and outputs $\mathbf{y}$:

$$\mathbf{x} = [\dot{m}_{evap}, T_{evap,e}, T_{evap,b}, T_{evap,l}, T_{cond,e}, T_{cond,b}, T_{cond,l}]^T, \quad \mathbf{u} = [T_{evap,b}^{set}, T_{evap,l}^{set}, T_{cond,e}^{set}]^T, \\ \mathbf{d} = [T_{db}, RH, \dot{Q}_{load}]^T, \quad \mathbf{y} = [PLR_1, P_1, COP_1, PLR_2, P_2, COP_2, P_{ahu}, P_{ct}, P_{p,evap}]^T.$$

The equations governing each modeling approach are given below. Once more, $\mathcal{N}$ represents a Multi-Layer Perceptron (MLP) with parameters $\boldsymbol{\omega}$.

- Black-box non-component-based:

$$\begin{cases} \dot{\mathbf{x}} = \mathcal{N}_1(\boldsymbol{\xi}; \boldsymbol{\omega}_1), \\ \mathbf{y} = \mathcal{N}_2(\boldsymbol{\xi}; \boldsymbol{\omega}_2), \end{cases}$$

$$\text{where } \boldsymbol{\xi} = \begin{bmatrix} \mathbf{x} \\ \mathbf{u} \\ \mathbf{d} \end{bmatrix}.$$

- Black-box component-based:

$$\dot{\mathbf{x}} = \begin{pmatrix} \ddot{m}_{evap} \\ \dot{T}_{evap,e} \\ \dot{T}_{evap,b} \\ \dot{T}_{evap,l} \\ \dot{T}_{cond,e} \\ \dot{T}_{cond,b} \\ \dot{T}_{cond,l} \end{pmatrix} = \begin{pmatrix} \mathcal{N}_1(\boldsymbol{\xi}_{pump}; \boldsymbol{\omega}_1) \\ \mathcal{N}_2(\boldsymbol{\xi}_{IT}; \boldsymbol{\omega}_2) \\ \mathcal{N}_3^{(1)}(\boldsymbol{\xi}_{ch1}; \boldsymbol{\omega}_3) \\ \mathcal{N}_4^{(1)}(\boldsymbol{\xi}_{ch2}; \boldsymbol{\omega}_4) \\ \mathcal{N}_5(\boldsymbol{\xi}_{ct}; \boldsymbol{\omega}_5) \\ \mathcal{N}_3^{(2)}(\boldsymbol{\xi}_{ch1}; \boldsymbol{\omega}_3) \\ \mathcal{N}_4^{(2)}(\boldsymbol{\xi}_{ch2}; \boldsymbol{\omega}_4) \end{pmatrix}, \quad \mathbf{y} = \begin{pmatrix} P_{p,evap} \\ P_{ahu} \\ PLR_1 \\ P_1 \\ COP_1 \\ P_{ct} \\ PLR_2 \\ P_2 \\ COP_2 \end{pmatrix} = \begin{pmatrix} \mathcal{N}_6(\boldsymbol{\xi}_{pump}; \boldsymbol{\omega}_6) \\ \mathcal{N}_7(\boldsymbol{\xi}_{IT}; \boldsymbol{\omega}_7) \\ \mathcal{N}_8^{(1)}(\boldsymbol{\xi}_{ch1}; \boldsymbol{\omega}_8) \\ \mathcal{N}_8^{(2)}(\boldsymbol{\xi}_{ch1}; \boldsymbol{\omega}_8) \\ \mathcal{N}_8^{(3)}(\boldsymbol{\xi}_{ch1}; \boldsymbol{\omega}_8) \\ \mathcal{N}_9(\boldsymbol{\xi}_{ct}; \boldsymbol{\omega}_9) \\ \mathcal{N}_{10}^{(1)}(\boldsymbol{\xi}_{ch2}; \boldsymbol{\omega}_{10}) \\ \mathcal{N}_{10}^{(2)}(\boldsymbol{\xi}_{ch2}; \boldsymbol{\omega}_{10}) \\ \mathcal{N}_{10}^{(3)}(\boldsymbol{\xi}_{ch2}; \boldsymbol{\omega}_{10}) \end{pmatrix}.$$

where

$$\boldsymbol{\xi}_{ch1} = [T_{cond,b}, T_{cond,l}, T_{evap,e}, T_{evap,b}, T_{evap,l}^{set}, \dot{m}_{evap}]^T,$$

$$\boldsymbol{\xi}_{ch2} = [T_{cond,e}, T_{cond,b}, T_{evap,b}, T_{evap,l}, T_{evap,l}^{set}, \dot{m}_{evap}]^T,$$

$$\xi_{pump} = [T_{evap,l}, \dot{m}_{evap}, \dot{Q}_{load}]^T, \quad \xi_{IT} = [T_{evap,e}, T_{evap,l}, \dot{m}_{evap}, \dot{Q}_{load}]^T$$

and

$$\xi_{ct} = [T_{db}, RH, T_{cond,l}, T_{cond,e}, T_{cond,e}^{set}]^T.$$

Analogously, $\mathcal{N}_i^{(j)}$ denotes the j th node of the output layer of the i th MLP. This notation will be used consistently throughout the remainder of the section. where

- Gray-box non-component-based:

$$\dot{\mathbf{x}} = \begin{pmatrix} \ddot{m}_{evap} \\ \dot{T}_{evap,e} \\ \dot{T}_{evap,b} \\ \dot{T}_{evap,l} \\ \dot{T}_{cond,e} \\ \dot{T}_{cond,b} \\ \dot{T}_{cond,l} \end{pmatrix} = \begin{pmatrix} \mathcal{N}_1^{(1)}(\xi; \omega_1) \\ a_0 \dot{Q}_{load} - a_1 T_{evap,e} + a_2 T_{evap,l} + a_3 + \mathcal{N}_1^{(2)}(\xi; \omega_1) \\ b_0 T_{evap,b}^{set} - b_1 T_{evap,b} + b_2 T_{evap,e} + b_3 + \mathcal{N}_1^{(3)}(\xi; \omega_1) \\ c_0 T_{evap,l}^{set} - c_1 T_{evap,l} + c_2 T_{evap,b} + c_3 + \mathcal{N}_1^{(4)}(\xi; \omega_1) \\ d_0 T_{cond,e}^{set} - d_1 T_{cond,e} + d_2 T_{cond,l} + d_3 + \mathcal{N}_1^{(5)}(\xi; \omega_1) \\ -e_0 T_{evap,l}^{set} - e_1 T_{cond,b} + e_2 T_{cond,e} + e_3 + \mathcal{N}_1^{(6)}(\xi; \omega_1) \\ -f_0 T_{evap,b}^{set} - f_1 T_{cond,l} + f_2 T_{cond,b} + f_3 + \mathcal{N}_1^{(7)}(\xi; \omega_1) \end{pmatrix},$$

$$\mathbf{y} = \begin{pmatrix} P_{p,evap} \\ P_{ahu} \\ PLR_1 \\ P_1 \\ COP_1 \\ P_{ct} \\ PLR_2 \\ P_2 \\ COP_2 \end{pmatrix} = \begin{pmatrix} \mathcal{N}_2^{(1)}(\xi; \omega_2) \\ \mathcal{N}_2^{(2)}(\xi; \omega_2) \\ \frac{\dot{Q}_{evap}}{\alpha_1} + \mathcal{N}_2^{(3)}(\xi; \omega_2) \\ \text{ALAMO}(\beta_1, T_{cond,e}, T_{evap,l}, PLR_2) + \mathcal{N}_2^{(4)}(\xi; \omega_2) \\ \frac{\dot{Q}_{evap}}{P_1} + \mathcal{N}_2^{(5)}(\xi; \omega_2) \\ \mathcal{N}_2^{(6)}(\xi; \omega_2) \\ \frac{\dot{Q}_{evap}}{\alpha_2} + \mathcal{N}_2^{(7)}(\xi; \omega_2) \\ \text{ALAMO}(\beta_2, T_{cond,e}, T_{evap,l}, PLR_2) + \mathcal{N}_2^{(8)}(\xi; \omega_2) \\ \frac{\dot{Q}_{evap}}{P_2} + \mathcal{N}_2^{(9)}(\xi; \omega_2) \end{pmatrix}.$$

- Gray-box component-based:

$$\dot{\mathbf{x}} = \begin{pmatrix} \ddot{m}_{evap} \\ \dot{T}_{evap,e} \\ \dot{T}_{evap,b} \\ \dot{T}_{evap,l} \\ \dot{T}_{cond,e} \\ \dot{T}_{cond,b} \\ \dot{T}_{cond,l} \end{pmatrix} = \begin{pmatrix} \mathcal{N}_1(\xi_{pump}; \omega_1) \\ a_0 \dot{Q}_{load} - a_1 T_{evap,e} + a_2 T_{evap,l} + a_3 + \mathcal{N}_2(\xi_{IT}; \omega_2) \\ b_0 T_{evap,b}^{set} - b_1 T_{evap,b} + b_2 T_{evap,e} + b_3 + \mathcal{N}_3^{(1)}(\xi_{ch1}; \omega_3) \\ c_0 T_{evap,l}^{set} - c_1 T_{evap,l} + c_2 T_{evap,b} + c_3 + \mathcal{N}_4^{(1)}(\xi_{ch2}; \omega_4) \\ d_0 T_{cond,e}^{set} - d_1 T_{cond,e} + d_2 T_{cond,l} + d_3 + \mathcal{N}_5(\xi_{ct}; \omega_5) \\ -e_0 T_{evap,l}^{set} - e_1 T_{cond,b} + e_2 T_{cond,e} + e_3 + \mathcal{N}_3^{(2)}(\xi_{ch1}; \omega_3) \\ -f_0 T_{evap,b}^{set} - f_1 T_{cond,l} + f_2 T_{cond,b} + f_3 + \mathcal{N}_4^{(2)}(\xi_{ch2}; \omega_4) \end{pmatrix},$$

$$\mathbf{y} = \begin{pmatrix} P_{p, \text{evap}} \\ P_{ahu} \\ PLR_1 \\ P_1 \\ COP_1 \\ P_{ct} \\ PLR_2 \\ P_2 \\ COP_2 \end{pmatrix} = \begin{pmatrix} \mathcal{N}_6(\boldsymbol{\xi}_{\text{pump}}; \boldsymbol{\omega}_6) \\ \mathcal{N}_7(\boldsymbol{\xi}_{IT}; \boldsymbol{\omega}_7) \\ \frac{\dot{Q}_{\text{evap}}}{\alpha_1} + \mathcal{N}_8^{(1)}(\boldsymbol{\xi}_{ch1}; \boldsymbol{\omega}_8) \\ \text{ALAMO}(\boldsymbol{\beta}_1, T_{\text{cond}, e}, T_{\text{evap}, l}, PLR_2) + \mathcal{N}_8^{(2)}(\boldsymbol{\xi}_{ch1}; \boldsymbol{\omega}_8) \\ \frac{\dot{Q}_{\text{evap}}}{P_1} + \mathcal{N}_8^{(3)}(\boldsymbol{\xi}_{ch1}; \boldsymbol{\omega}_8) \\ \mathcal{N}_9(\boldsymbol{\xi}_{ct}; \boldsymbol{\omega}_9) \\ \frac{\dot{Q}_{\text{evap}}}{\alpha_2} + \mathcal{N}_{10}^{(1)}(\boldsymbol{\xi}_{ch2}; \boldsymbol{\omega}_{10}) \\ \text{ALAMO}(\boldsymbol{\beta}_2, T_{\text{cond}, e}, T_{\text{evap}, l}, PLR_2) + \mathcal{N}_{10}^{(2)}(\boldsymbol{\xi}_{ch2}; \boldsymbol{\omega}_{10}) \\ \frac{\dot{Q}_{\text{evap}}}{P_2} + \mathcal{N}_{10}^{(3)}(\boldsymbol{\xi}_{ch2}; \boldsymbol{\omega}_{10}) \end{pmatrix}.$$

4.3 Model Training

Let the model of the system of interest be described by

$$\begin{cases} \dot{\mathbf{x}} = \hat{f}_x(\mathbf{x}, \mathbf{u}, \mathbf{d}; \boldsymbol{\omega}_x), \\ \mathbf{y} = \hat{f}_y(\mathbf{x}, \mathbf{u}, \mathbf{d}; \boldsymbol{\omega}_y), \end{cases}$$

where $\mathbf{x} \in \mathbb{R}^{n_x}$ are the system states, $\mathbf{y} \in \mathbb{R}^{n_y}$ the outputs, $\mathbf{u} \in \mathbb{R}^{n_u}$ the control inputs, and $\mathbf{d} \in \mathbb{R}^{n_d}$ the disturbances. The functions $\hat{f}_x$ and $\hat{f}_y$ represent the dynamical and output models, respectively, and can take any of the forms detailed in [Subsection 4.2.1](#) and [Subsection 4.2.2](#).

Unlike standard state-space models, where only the outputs $\mathbf{y}$ are observed, in this project data is available for all relevant variables: the system states $\mathbf{x}$, control inputs $\mathbf{u}$, disturbances $\mathbf{d}$, and outputs $\mathbf{y}$. This enables direct supervision during training of both the dynamical and output models.

The training and validation datasets are generated using the ground-truth model using different input and disturbance profiles. As a result, each dataset captures distinct system trajectories. All datasets contain complete trajectories of $\mathbf{x}, \mathbf{u}, \mathbf{d}, \mathbf{y}$, sampled at fixed intervals of $\tau = 30$ seconds.

To avoid training biases caused by differences in variable magnitudes, all variables are normalized to have zero mean and unit standard deviation.

The training process is carried out in two stages. In the first stage, the parameters $\boldsymbol{\omega}_x$ of the Neural Ordinary Differential Equation (NODE), which governs the evolution of the state variables, are optimized following the procedure described in [Subsection 2.2.1](#). This stage uses the observed trajectories of $\mathbf{x}, \mathbf{u}$, and $\mathbf{d}$ from the training dataset to fit the model $\hat{f}_x$. The resulting parameter set $\boldsymbol{\omega}_x^*$ corresponds to the lowest validation loss achieved during training.

In the second stage, the parameters $\boldsymbol{\omega}_y$ of the Multi-Layer Perceptron(s) (MLPs) responsible for modeling the output function $\hat{f}_y$ are trained. It was observed that using the NODE-predicted states $\mathbf{x}$ led to more accurate output predictions than using the observed states $\hat{\mathbf{x}}$. Therefore, the NODE is used to generate predicted state trajectories for both the training and validation datasets. These predicted states, along with $\mathbf{u}$ and $\mathbf{d}$, are used to train the output model following the procedure in [Subsection 2.1.1](#), yielding the final parameter set $\boldsymbol{\omega}_y^*$.

4.4 Model Testing

After training, the model with the learned parameters ω_x^* and ω_y^* is evaluated on a separate test dataset. It contains unseen sequences of $\mathbf{x}, \mathbf{u}, \mathbf{d}, \mathbf{y}$ sampled at fixed intervals of $\tau = 30$ seconds that were not used during training or validation. The NODE is provided with the initial condition of the system from the test dataset and integrated forward in time using the Explicit Euler method to simulate the evolution of the state variables $\mathbf{x}$, which are then fed into the function $\hat{f}_y$ to obtain the predicted outputs $\hat{\mathbf{y}}$.

To assess the model's predictive accuracy, two key performance indicators (KPIs) are computed: the Relative Root Mean Square Error (RRMSE) and the Maximum Absolute Error (MAE). The RRMSE is used to quantify the overall accuracy of the predictions, while the MAE highlights the worst-case prediction errors across the test set.

For a vector $\mathbf{z}^{(i)} = [z_t^{(i)}]_{t=0}^T$ containing the predicted time evolution of a variable $z^{(i)}$ for $i = \{1, \dots, n_z\}$ and $z = \{x, y\}$, these metrics are defined as

$$\text{RRMSE}(\mathbf{z}^{(i)}) = \frac{1}{\bar{z}^{(i)}} \sqrt{\frac{\|\mathbf{z}^{(i)} - \hat{\mathbf{z}}^{(i)}\|_2^2}{T}}$$

and

$$\text{MAE}(\mathbf{z}^{(i)}) = \|\mathbf{z}^{(i)} - \hat{\mathbf{z}}^{(i)}\|_\infty.$$

The reference values $\bar{z}^{(i)}$ are collected in [Table 4.1](#).

Table 4.1: Reference values of the variables used to compute the RRMSE.

Variable	Reference Value
$T_{evap,e}$	20°C
$T_{evap,b}$	20°C
$T_{evap,l}$	20°C
$T_{cond,e}$	20°C
$T_{cond,b}$	20°C
$T_{cond,l}$	20°C
$\dot{m}_{evap}$	60 kg/s
P_1	150 kW
P_2	150 kW
P_{ct}	15 kW
P_{ahu}	20 kW
$P_{p,evap}$	40 kW
COP_1	6
COP_2	6
PLR_1	0.5
PLR_2	0.5

To determine them, a visual inspection of each time series was conducted to identify its typical range, a rounded value near the center of this range was selected. Attending

to Figure 4.2, the reference values for $\dot{m}_{evap}$ and PLR_1 could be 60 kg/s and 150 kW, respectively. This approach ensures that the RRMSE values across different variables are of comparable magnitudes, thereby minimizing the risk of misinterpretation in the presentation of results.

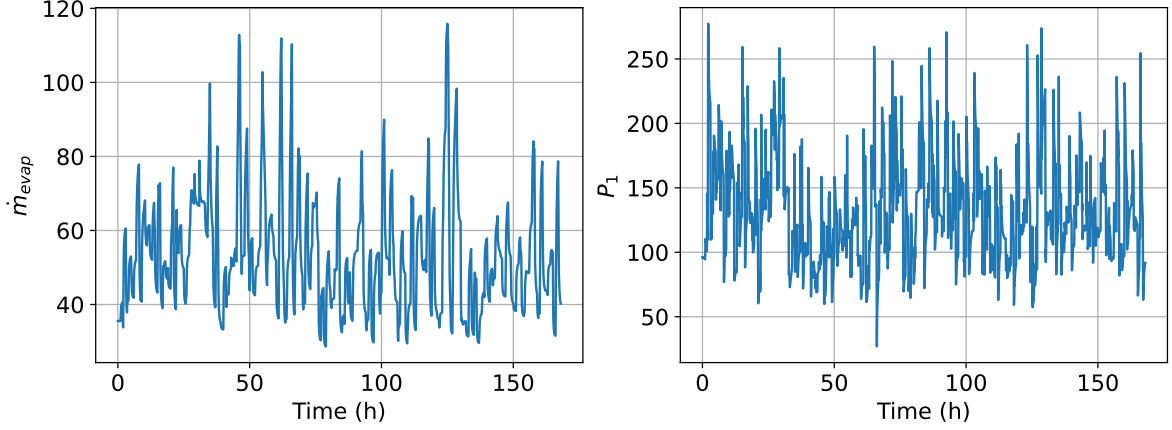


Figure 4.2: Time evolution over one week of the mass flow rate of the chilled water $\dot{m}_{evap}$ and the power consumption of the first chiller P_1 .

4.5 Set up for the Numerical Experiments

This section collects all the hyperparameters and settings that are unchanged from one experiment to another. Explicit Euler for the NODE

Training Parameter	NODEs (State Prediction)	MLPs (Output Prediction)
ODE solver	Explicit Euler	—
Optimizer	Adam	Adam
Learning rate (ϵ)	0.001	0.001
Number of training epochs	1000	200
Mini-batch size	10 samples	10 samples
Activation function	ReLU	ELU
Sampling time (τ)	30 seconds	30 seconds

Table 4.2: Training configuration for NODEs and MLPs.

Chapter 5

Numerical Results

5.1 Two-Chiller System

To evaluate whether the methodology developed in this thesis can effectively model HVAC applications, we consider a simplified system consisting of two isolated chillers. This reduced setup allows for an analysis of the model without the complexity of the complete chiller plant but keeping interesting dynamics.

To achieve this, the system will be modeled using all four modeling strategies. The datasets used for training and validation consist of three days of operation, with setpoints following a random step function as described in [Section 4.1](#). The testing dataset employs the same setpoint profile, extended over seven days of operation.

Once all the models described in [Subsection 4.2.1](#) are trained, the key performance indicators (KPIs) are computed on the test dataset. The results are presented in [Figure 5.1](#). It can be observed that all models achieve similar levels of predictive accuracy, as indicated by the close proximity of the markers corresponding to each variable.

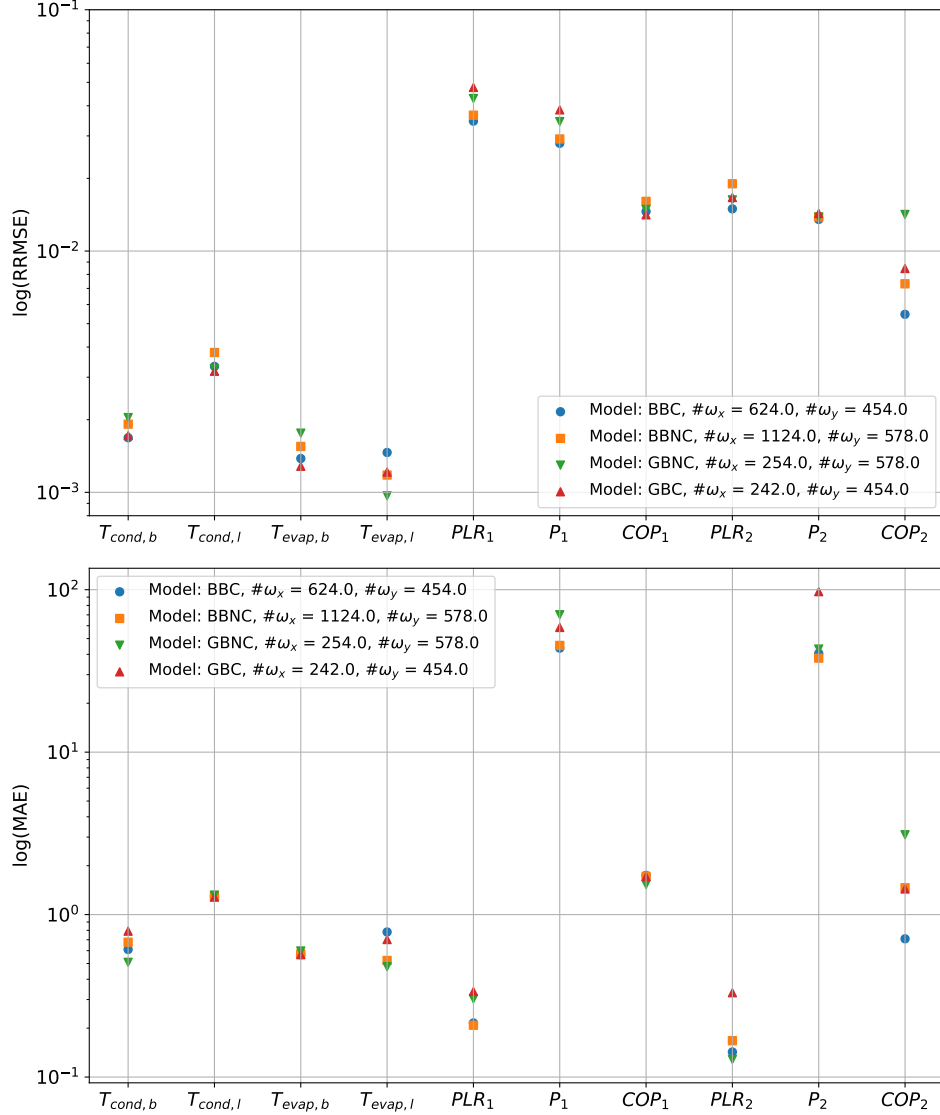


Figure 5.1: Comparison of the Relative Root Mean Squared Error (top) and Maximum Absolute Error (bottom) across all predicted variables for each modeling strategy. The models are categorized as Black-Box Component-based (BBC), Black-Box Non-Component-based (BBNC), Gray-Box Component-based (GBC), Gray-Box Non-Component-based (GBNC).

The prediction of the water temperatures at various locations within the system is significantly more accurate than that of the output variables. Specifically, the Relative Root Mean Squared Error (RRMSE) for all temperature predictions remains below 1%, while the RRMSE for the outputs is approximately an order of magnitude higher. This could be because the predictions of the outputs are obtained from the predicted states, therefore propagating the error.

A key distinction among the models lies in their complexity, which is quantified by the number of trainable parameters. Adopting a component-based formulation, where each subsystem is modeled separately, leads to a reduction in parameter count without de-

grading prediction accuracy, for both black-box and gray-box models. Moreover, incorporating physical knowledge into the formulation of Neural Ordinary Differential Equations (NODEs) further reduces model complexity without compromising performance. However, the same effect is not observed when introducing physics to model the relationship between system states and output variables.

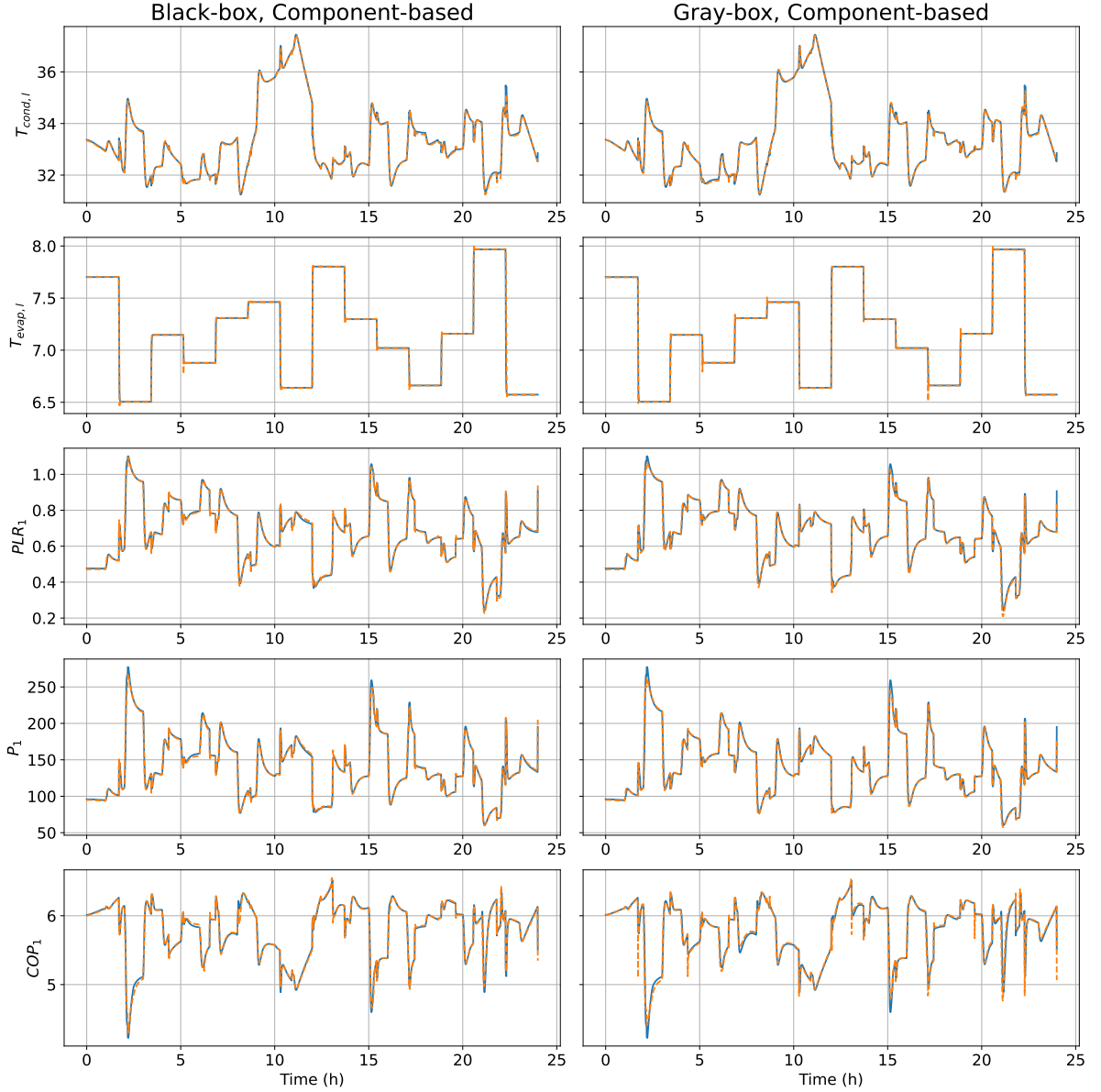


Figure 5.2: Comparison of the predicted trajectories for selected state and output variables using the test dataset for the first 24 hours. The left column shows the predictions produced by the Black-Box Component-based (BBC) model, while the right column shows those from the Gray-Box Component-based (GBC) model. In all plots, the true values are represented by a solid blue line, and the model predictions are shown as dashed orange lines.

Figure 5.2 shows that the predictions produced by both models closely follow the observed values, which is consistent with the low RRMSE and maximum absolute error reported earlier. Additionally, Figure 8.1 illustrates the prediction for the complete dataset for these variables.

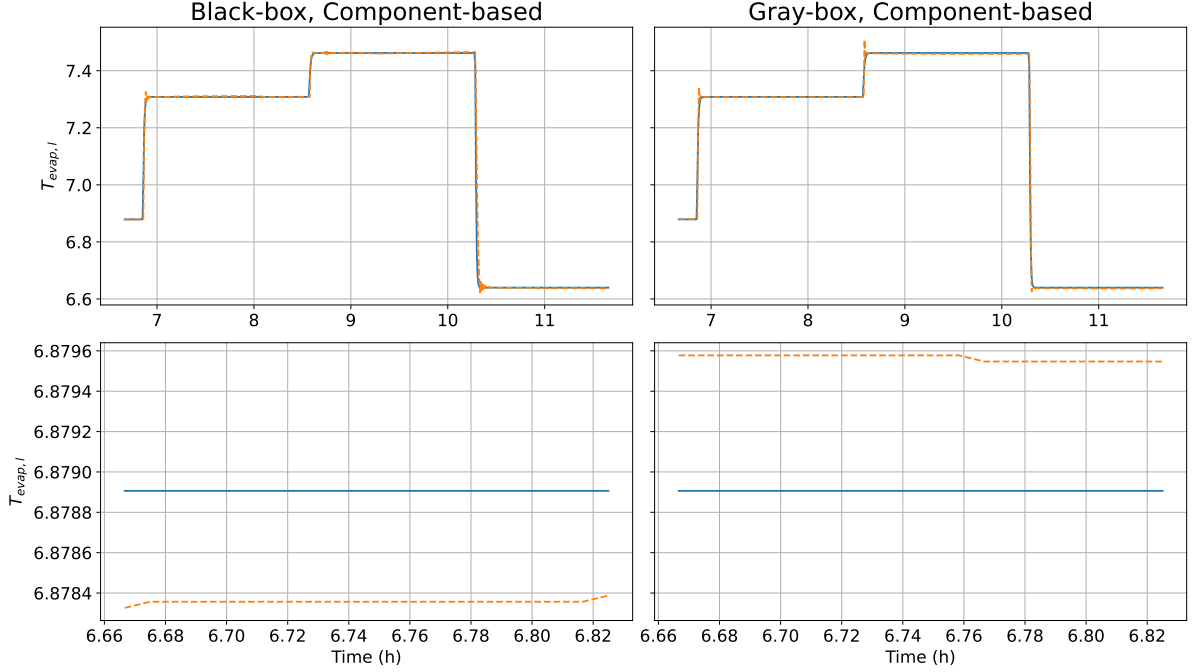


Figure 5.3: Zoomed-in view of the predicted and observed trajectories for the temperature of the water leaving the evaporator of the second chiller during a slow transient between two steady-state periods. The left and right panels show the predictions produced by the Black-Box Component-based (BBC) and Gray-Box Component-based (GBC) models, respectively. The observed values are represented by a solid blue line, while the model predictions are shown as dashed orange lines.

Figure 5.3 shows that both models capture the steady states and slow transients with high accuracy. However, small spikes are observed at the end of some transients, just before the system reaches the next steady state. This behavior is likely caused by the use of the Explicit Euler integration scheme with a time step that is too large, resulting in numerical oscillations over a few time steps.

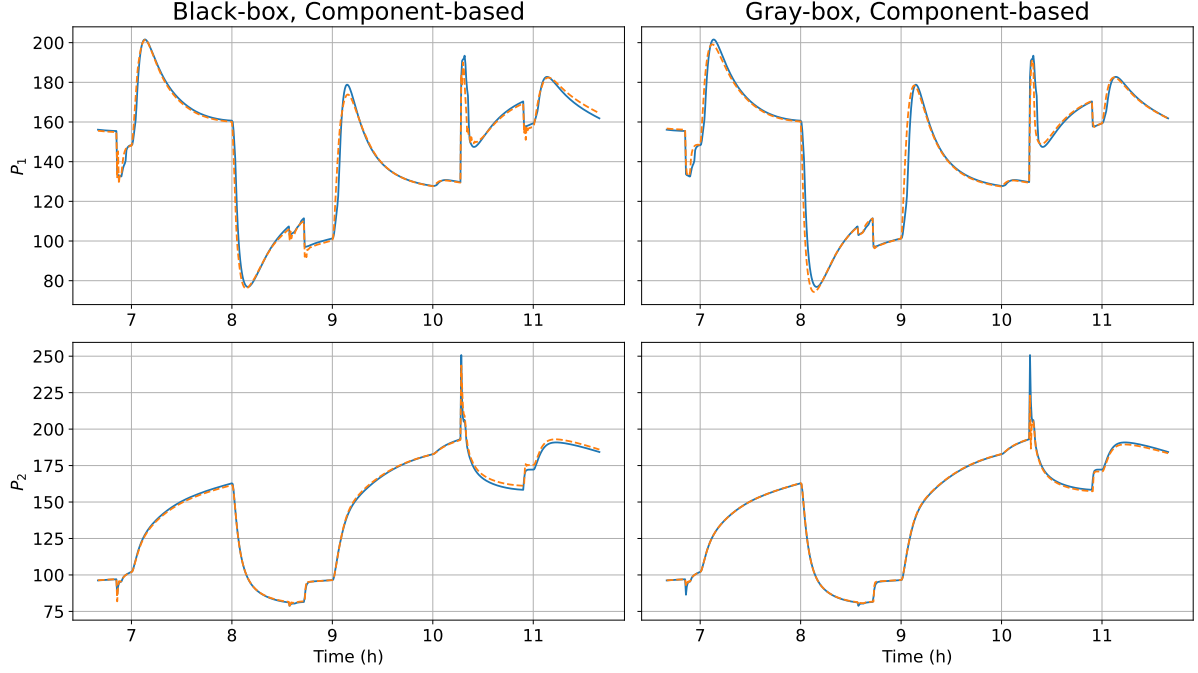


Figure 5.4: Zoomed-in view of the predicted and observed trajectories for the power consumption of both chillers. The left and right panels show the predictions produced by the Black-Box Component-based (BBC) and Gray-Box Component-based (GBC) models, respectively. The observed values are represented by a solid blue line, while the model predictions are shown as dashed orange lines.

Figure 5.4 shows that both models capture the time evolution of the power consumption of both chillers with high accuracy.

5.2 Complete HVAC System

5.2.1 Analysis of Model Structure

This experiment has two primary objectives. The first is to evaluate the impact of adopting a component-based modeling approach compared to a non-component-based one. The second is to assess whether incorporating physics into the modeling process improves predictive performance. To investigate these questions, three different network sizes were tested for each of the four modeling strategies.

All datasets used for training and validation comprised three days of operation, with setpoints and IT load following a random step function. The testing dataset employed the same profiles for both the setpoints and the IT load, extended to a seven-day operational period. This ensures that all variables remain within comparable ranges, thereby avoiding extrapolation scenarios, which are addressed later in the report.

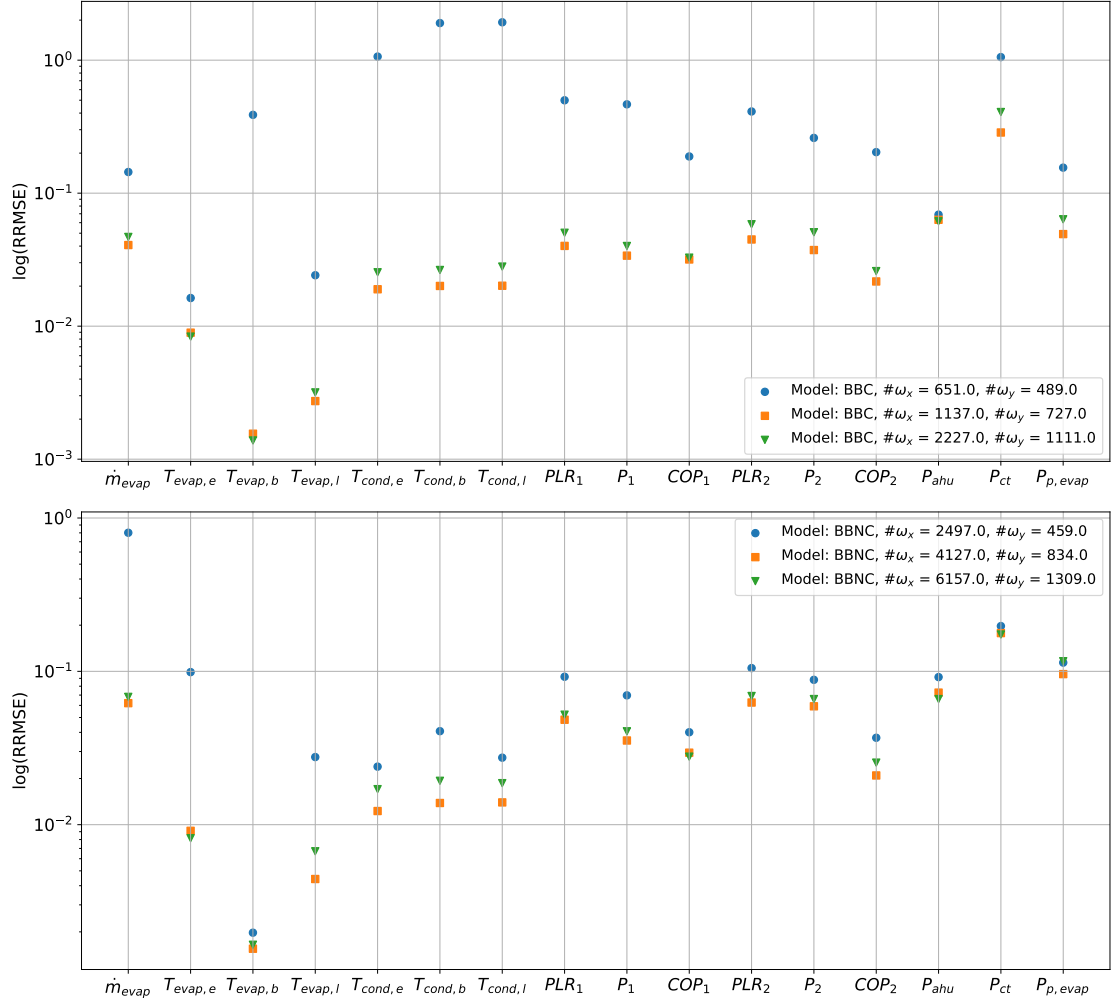


Figure 5.5: Relative Root Mean Squared Error (RRMSE) for all predicted variables using black-box component-based (top) and black-box non-component-based (bottom) modeling strategies for different model sizes.

Figure 5.5 shows that both the black-box component-based and non-component-based approaches yield the best prediction for the medium-sized network. Increasing or decreasing the number of parameters beyond this optimal point results in degraded performance on the test dataset. This behavior is consistent with expectations: models with too few parameters may lack the capacity to capture the underlying system dynamics, while excessively complex models are more susceptible to overfitting, thereby reducing their ability to generalize to unseen data.

It can also be observed that the component and non-component based approaches yield similar results with the difference that the former is more memory efficient as it requires less parameters.

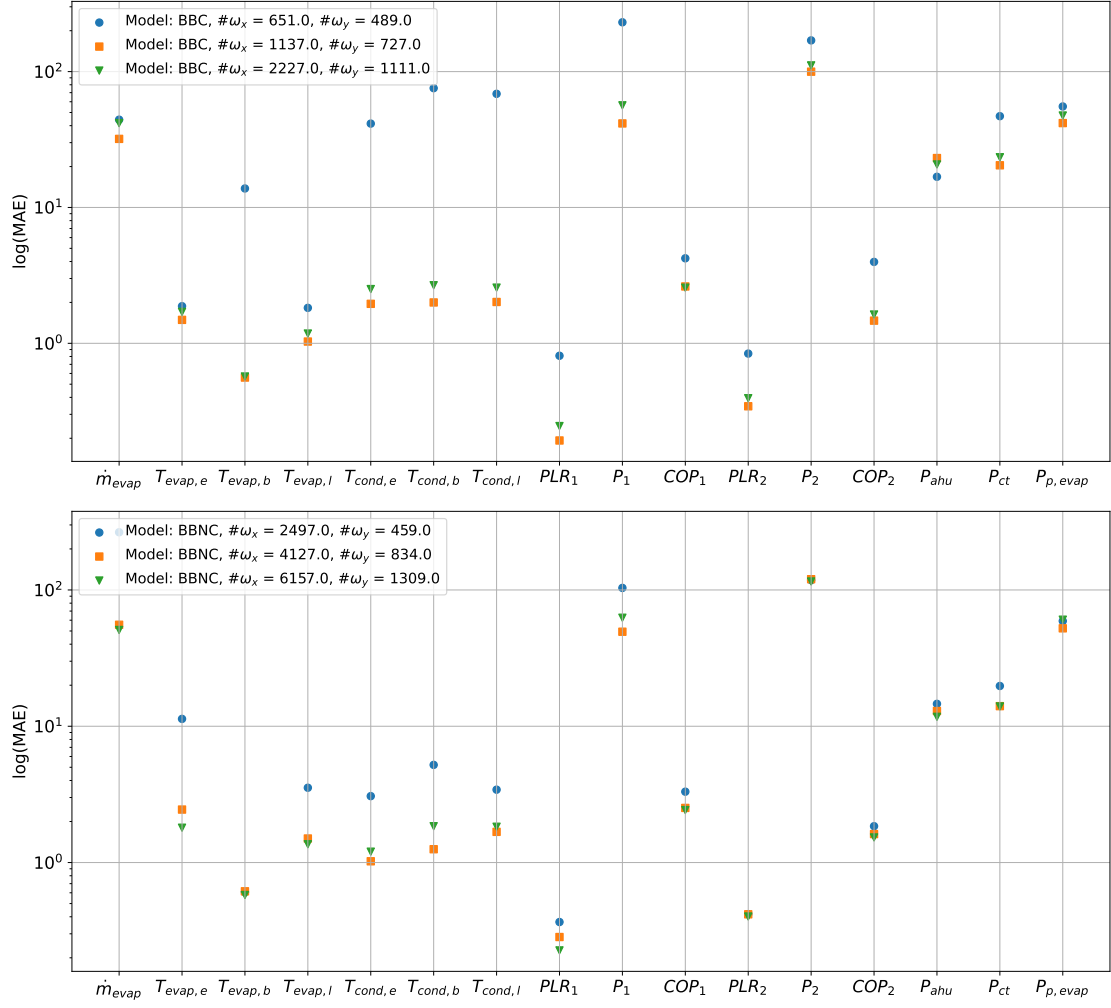


Figure 5.6: Maximum Absolute Error (MAE) for all predicted variables using black-box component-based (top) and black-box non-component-based (bottom) modeling strategies for different model sizes.

This behavior regarding the number of parameters can also be observed in Figure 5.6 for the component-based approach. However, for the non-component-based structure the difference between the medium and large models is smaller. This may occur because the possible overparametrization affects more significantly the average prediction performance of the model (RRMSE) than the worst prediction, whose error is given by the maximum absolute error.

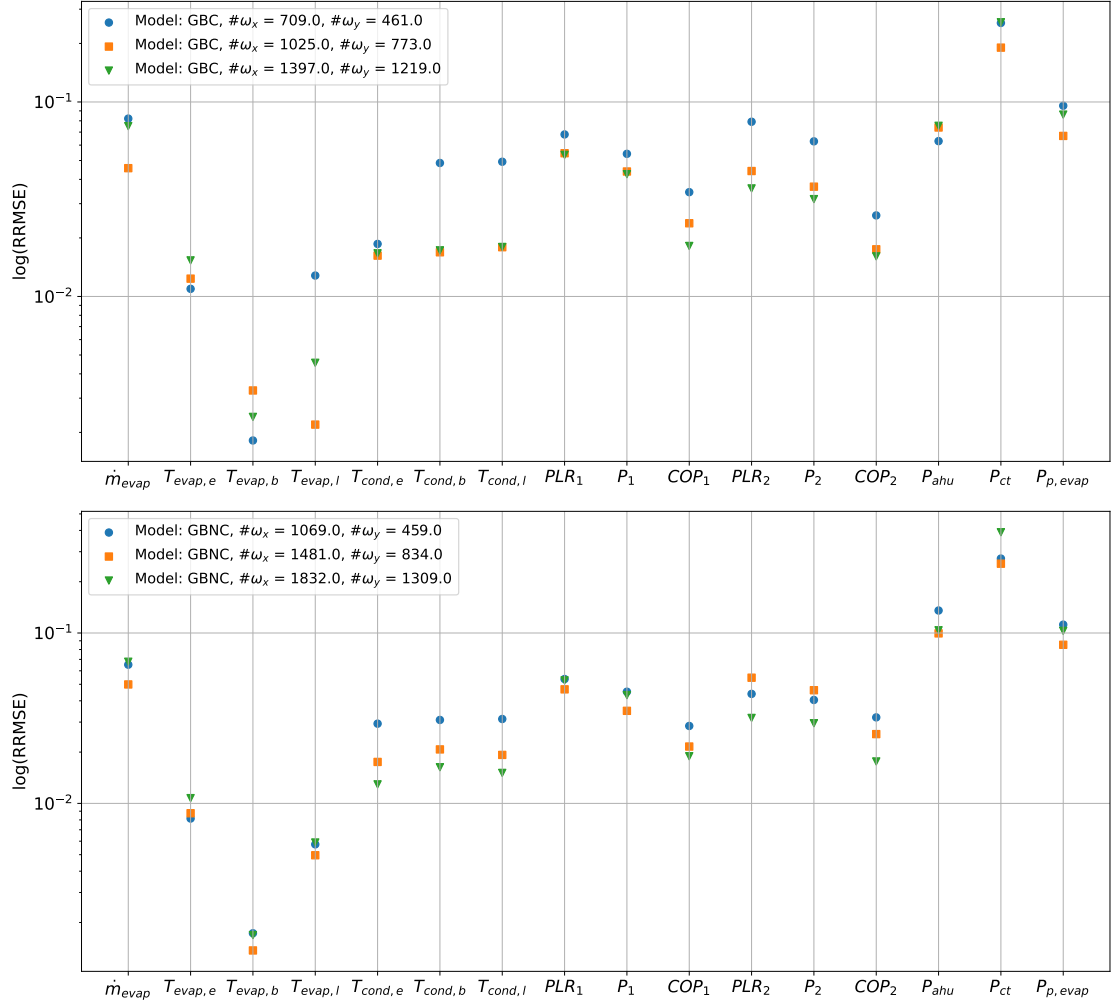


Figure 5.7: Relative Root Mean Squared Error (RRMSE) for all predicted variables using gray-box component-based (top) and gray-box non-component-based (bottom) modeling approaches for different model sizes.

On the other hand, [Figure 5.7](#) shows that the gray-box models do not exhibit the same sensitivity to the number of parameters. Increasing the network size does not significantly deteriorate the performance, which may be attributed to the regularizing effect of the physics-based equations. These constraints guide the learning process, making the models more robust to overfitting even when their expressive capacity is increased.

Additionally, the values of the RRMSE and the maximum absolute error presented in [Figure 5.7](#) indicate that both strategies have similar prediction accuracies. Analogously to what occurred with the black-box models, the component-based approach is more memory efficient than the alternative.

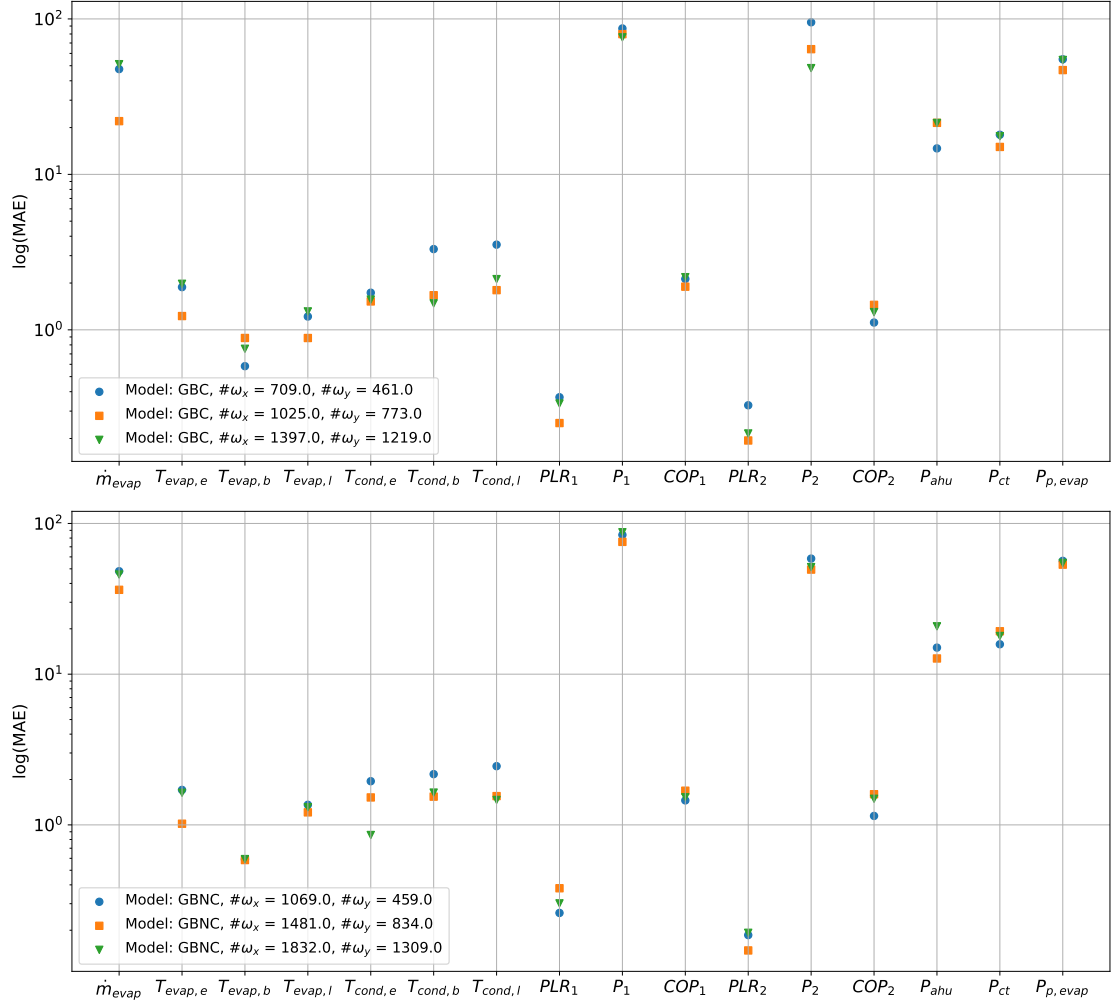


Figure 5.8: Maximum Absolute Error (MAE) for all predicted variables using gray-box component-based (top) and gray-box non-component-based (bottom) modeling approaches for different model sizes.

The same phenomenon is illustrated in Figure 5.8, where both gray-box models provide similar maximum absolute values regardless of the number of parameters.

The analysis of how model size influences predictive performance across the four modeling strategies has shown that medium-sized models consistently outperform configurations with lower complexity and achieve results comparable to, or in some cases better than, the larger models while requiring significantly fewer parameters. For this reason, the medium-sized models are selected for comparison to determine which strategy provides the most accurate predictions.

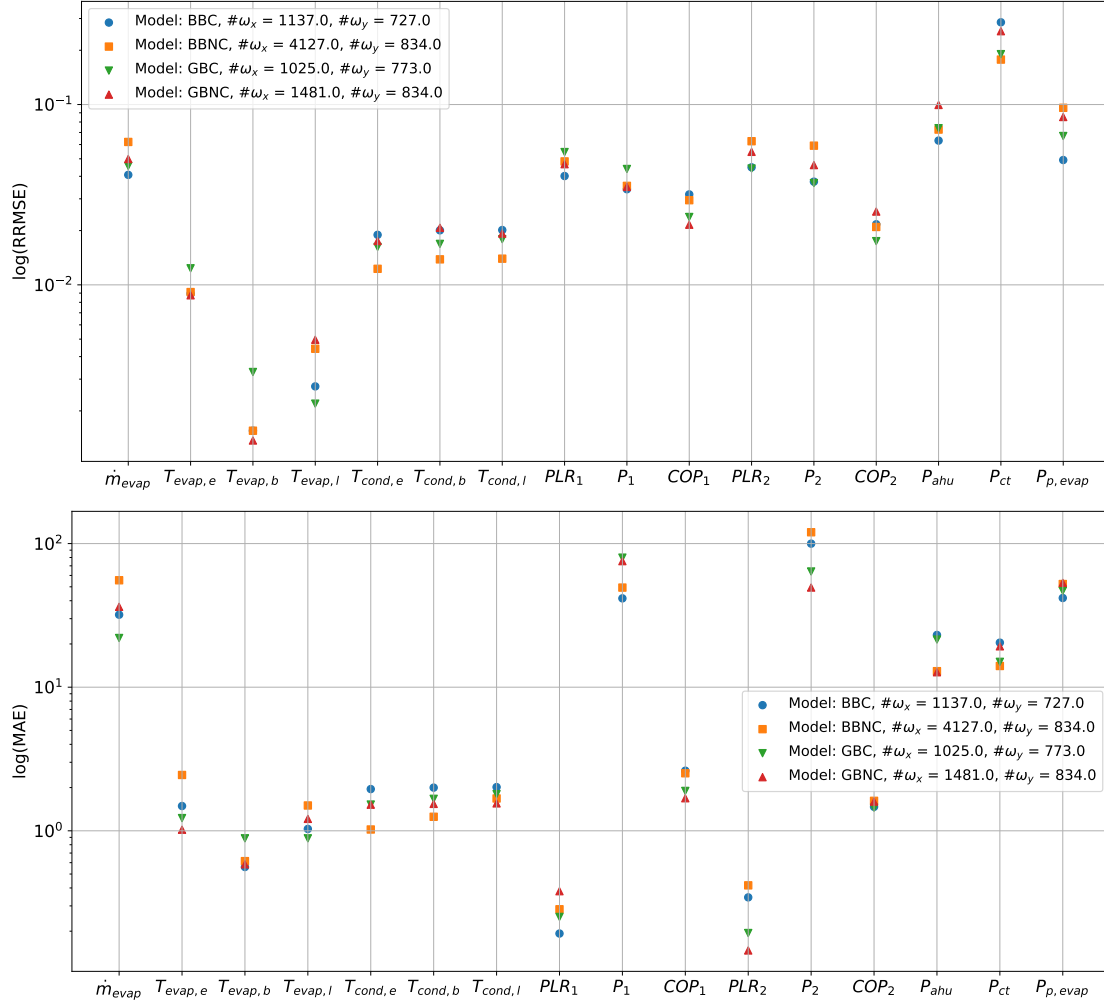


Figure 5.9: Comparison of Relative Root Mean Squared Error (top) and Maximum Absolute Error (bottom) across all predicted variables for the four modeling strategies: Black-Box Component-based (BBC), Black-Box Non-Component-based (BBNC), Gray-Box Component-based (GBC), and Gray-Box Non-Component-based (GBNC). The results correspond to the selected optimal network size for each approach.

Figure 5.9 shows that the four strategies achieve very similar levels of predictive accuracy for all the variables. However, it is also evident that the RRMSE values are higher compared to the case of the isolated two-chiller system. This degradation in performance can be attributed to the fact that, in the full chiller plant scenario, the temperatures of the chilled and cooling water entering the chillers are no longer treated as inputs but are instead predicted by the model. Consequently, any error in the predicted entering water temperatures propagates through the system, also affecting the accuracy of the output variable predictions. Additionally, the increased dimensionality of the system introduces further complexity, making the learning process more challenging.

The results also indicate that the component-based approach requires fewer parameters than the non-component-based strategy to achieve comparable prediction performance, as

measured by the selected KPIs. Similarly to the two-chillers system, introducing physics in the Neural Ordinary Differential Equations (NODEs) reduces the number of parameters needed for similar accuracy. This behavior can be attributed to the fact that, in the component-based approach, each neural network is tasked with modeling a smaller and simpler subsystem. As a result, the benefit of incorporating first-order differential equations is less significant, since the learning task is already less complex.

In contrast, it can be observed that introducing physics to model the function that connects system states with the outputs does not present the same effect.

Additionally, the values of the KPIs illustrated in [Figure 5.9](#) show that the gray-box component based model is the best option to model a chiller plant among the other tested options as it provides similar values of the RRMSE and maximum absolute error while requiring the fewest number of parameters.

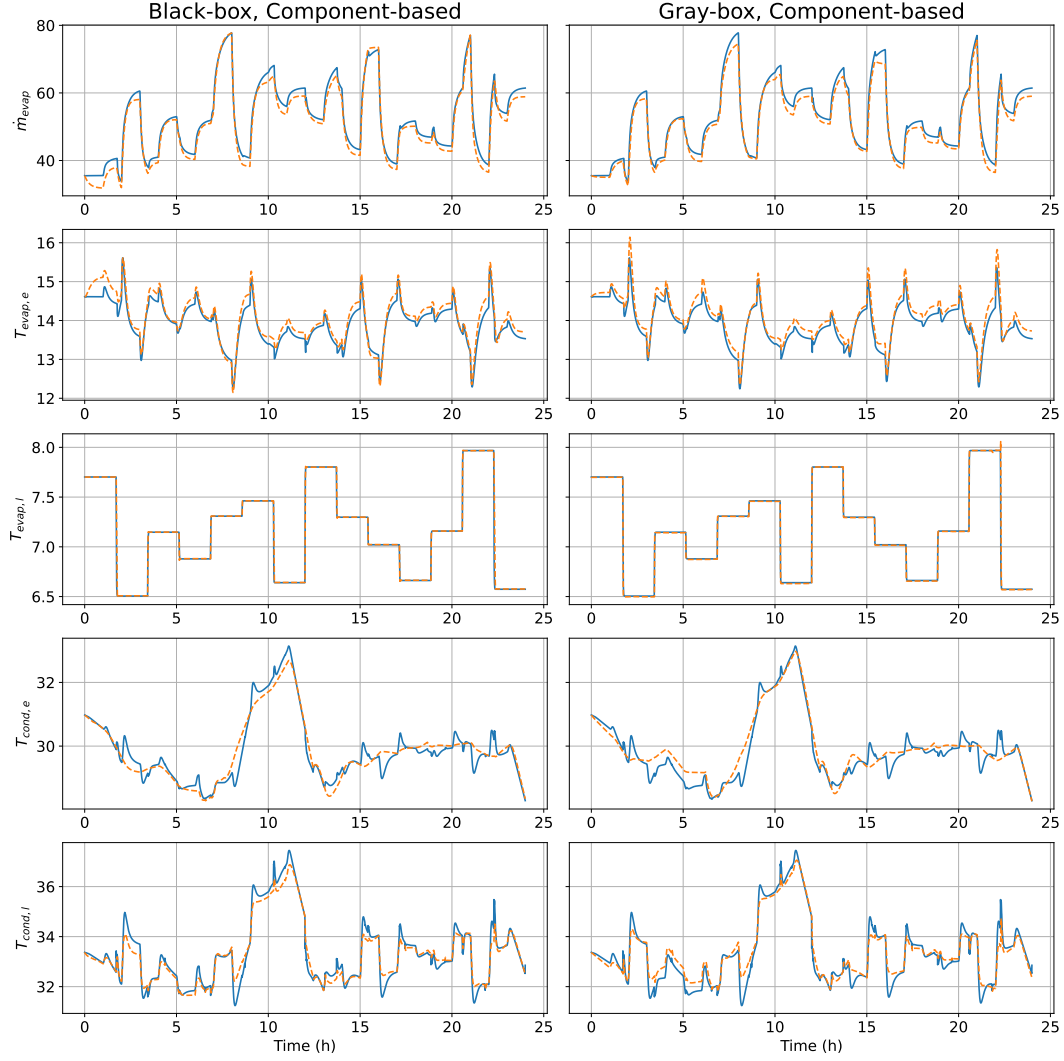


Figure 5.10: Comparison of the predicted trajectories for selected state variables using the test dataset for the first 24 hours. The left column shows the predictions produced by the Black-Box Component-based (BBC) model, while the right column shows those from the Gray-Box Component-based (GBC) model. In all plots, the true values are represented by a solid blue line, and the model predictions are shown as dashed orange lines.

The results shown in [Figure 5.10](#) confirm that both the Black-Box Component-based (BBC) and Gray-Box Component-based (GBC) models produce predictions that closely follow the observed trajectories of the state variables. This behavior is consistent with the low RRMSE values ($\text{RRMSE} < 2\%$) previously reported, indicating accurate overall performance. The plot illustrating the prediction of the same variables for the complete test dataset is [Figure 8.2](#).

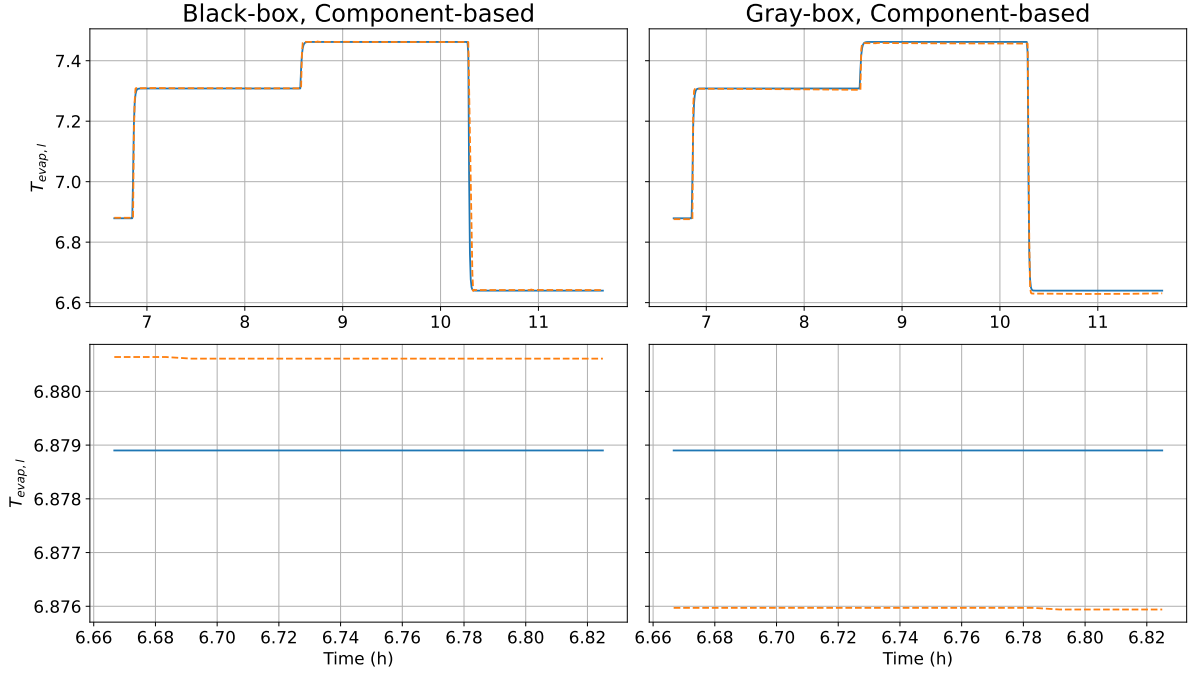


Figure 5.11: Zoomed-in view of the predicted and observed trajectories for the temperature of the water leaving the evaporator of the second chiller during a slow transient between two steady-state periods. The left and right panels show the predictions produced by the Black-Box Component-based (BBC) and Gray-Box Component-based (GBC) models, respectively. The observed values are represented by a solid blue line, while the model predictions are shown as dashed orange lines.

Regarding the steady-state behavior and the slow transients, [Figure 5.11](#) illustrates the fact that both strategies capture these phenomena with high accuracy. In contrast to the results obtained for the two isolated chillers, both black and gray-box models do not exhibit visible overshoots when the system reaches a new steady state.

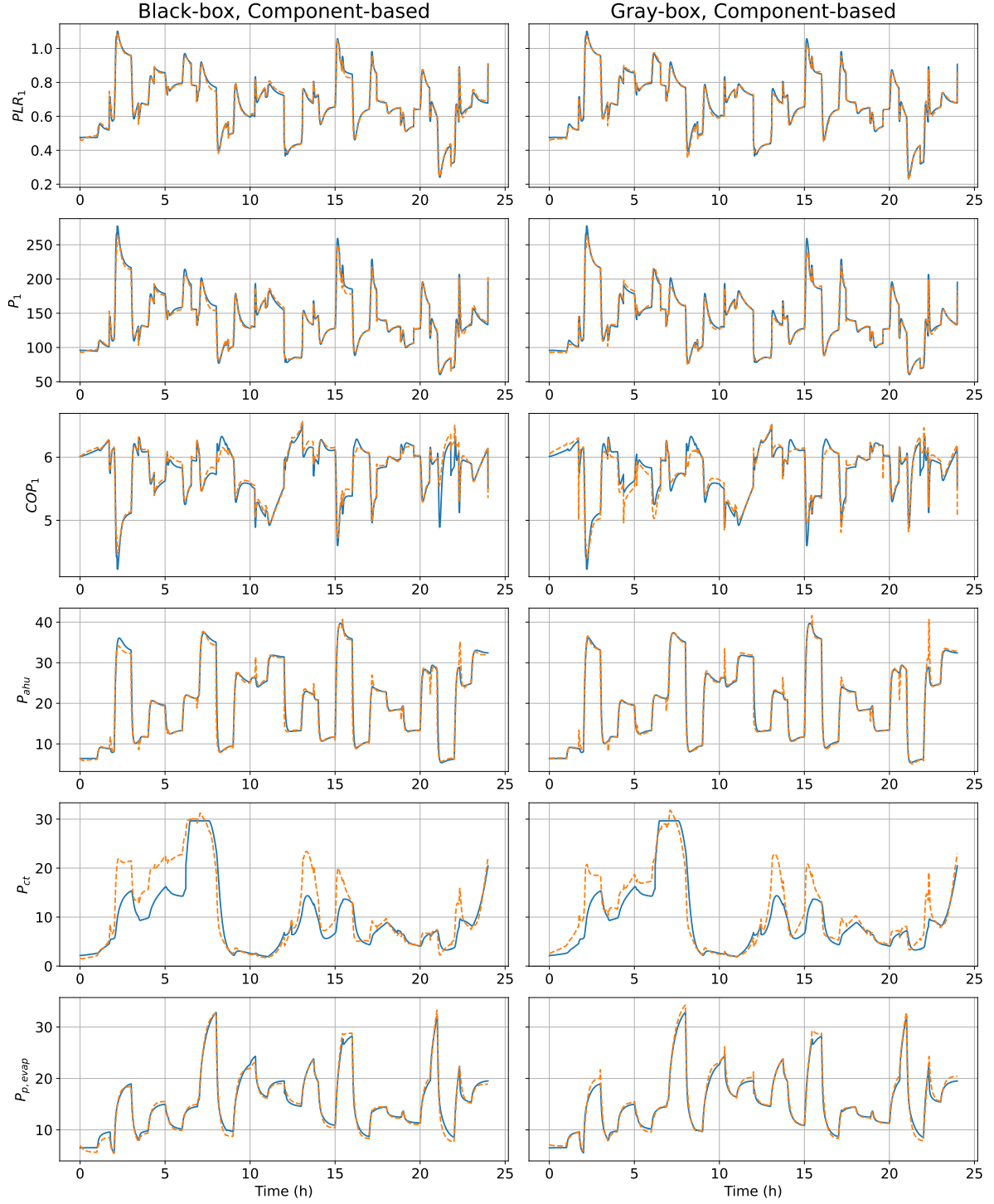


Figure 5.12: Comparison of the predicted trajectories for selected output variables using the test dataset for the first 24 hours. The left column shows the predictions produced by the Black-Box Component-based (BBC) model, while the right column shows those from the Gray-Box Component-based (GBC) model. In all plots, the true values are represented by a solid blue line while the model predictions are shown as dashed orange lines.

Figure 5.12 shows the predictions for the output variables for both gray-box component-based and black-box component-based. In particular, it can be observed that the models prediction follow closely the ground truth except for the power consumption of the cooling tower. Figure 8.3 illustrates the prediction of the the same output variables for the complete test dataset.

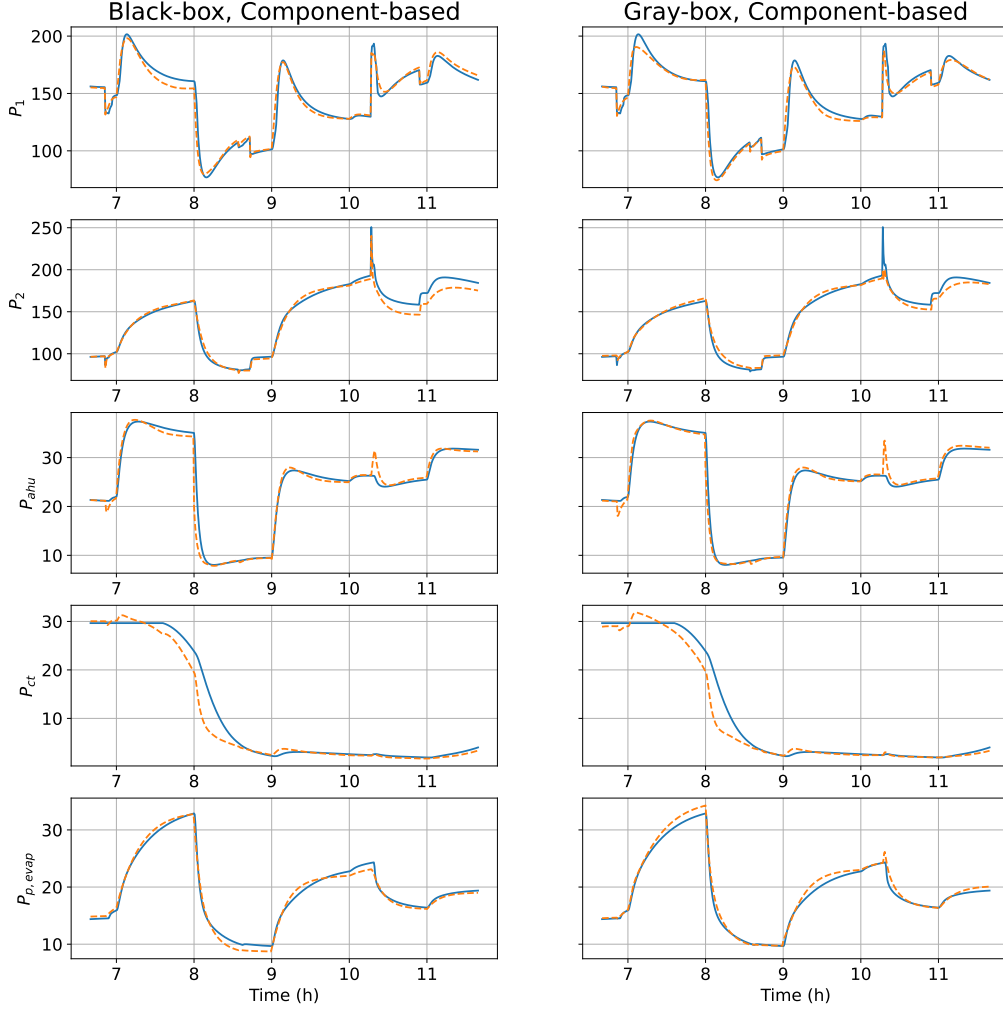


Figure 5.13: Zoomed-in view of the predicted and observed trajectories for the power consumption of all the components in the system. The left and right panels show the predictions produced by the Black-Box Component-based (BBC) and Gray-Box Component-based (GBC) models, respectively. The observed values are represented by a solid blue line, while the model predictions are shown as dashed orange lines.

Finally, Figure 5.13 illustrates that both models capture the relationship between the system states and the outputs with similar accuracy.

5.2.2 Study on the influence of Training Data Quantity

As previously mentioned, neural networks typically require large amounts of data to achieve accurate predictions, especially when modeling complex systems such as HVAC plants. In this section, we investigate whether incorporating physical knowledge into the model helps alleviate this dependency on large datasets. Additionally, we evaluate the impact of increasing the amount of training data on the predictive performance of the models.

While it is reasonable to expect that using more training data should lead to improved accuracy, this improvement must be weighed against the practical cost of data collection and model training. In particular, we assess whether the performance gain justifies the additional effort required to generate and process larger datasets.

To conduct this analysis, the gray-box component-based and black-box component-based models are used with the number of parameters that yielded the best results in the previous section. The models are trained and validated on datasets containing either 1, 3 and 7 days of data with setpoints and IT load following a random step function. This ensures consistent system operation and avoids extrapolation. The test dataset is the same as the one used in the previous subsection.

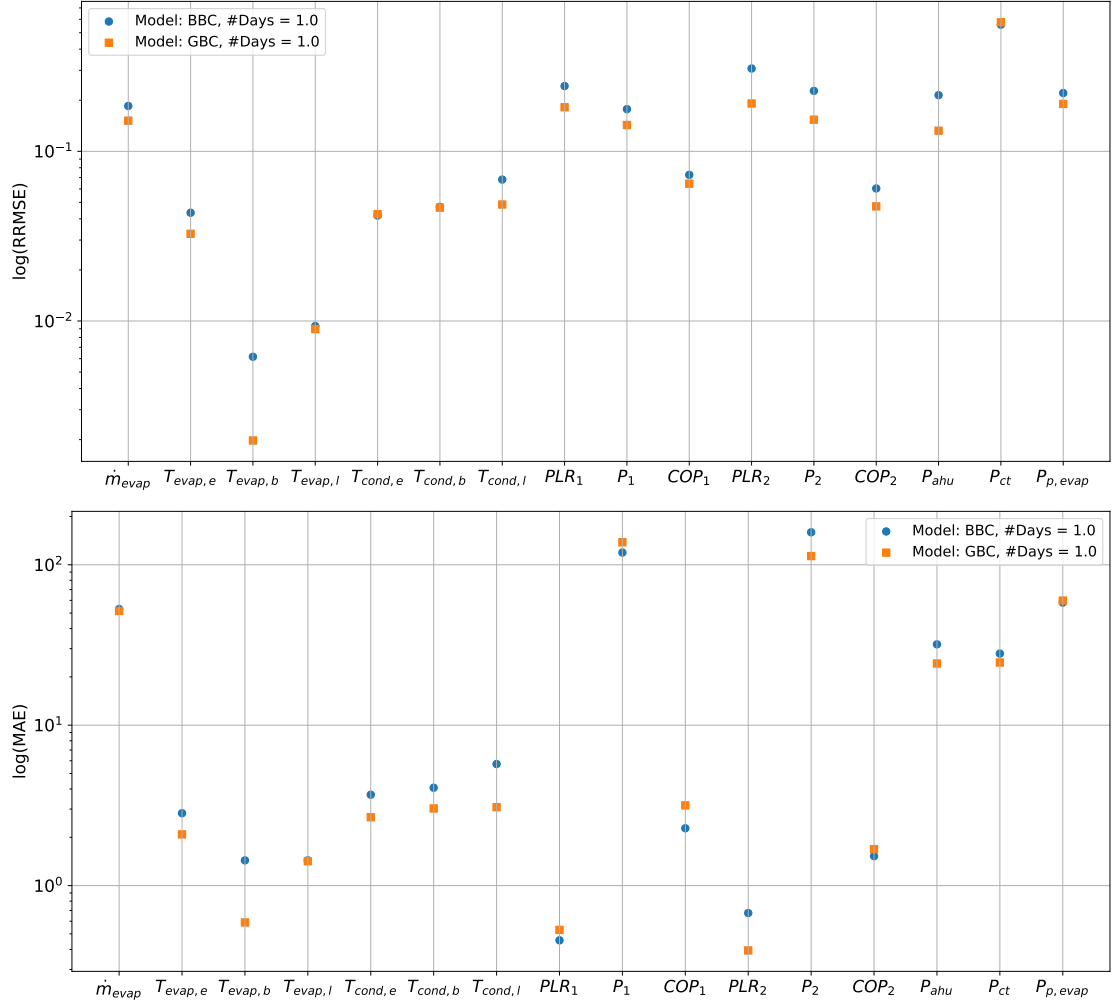


Figure 5.14: Comparison of Relative Root Mean Squared Error (top) and Maximum Absolute Error (bottom) across all predicted variables for the two modeling strategies: Black-Box Component-based (BBC) and Gray-Box Component-based (GBC). The results correspond to using data equivalent to one day for training.

Figure 5.14 indicates that both strategies achieve comparable performance, with the gray-box model showing slightly better results overall. Although the variables span similar ranges in both datasets, it has been observed that training both strategies on only one day of data may lead to unstable performance, depending on the initialization of the neural network parameters.

For the case presented in this report, the trained gray-box component-based model was unstable when computing the predictions of the systems's states for the training dataset. Consequently, the output estimations provided by the physics-based equations were computed directly from the observed input trajectories.

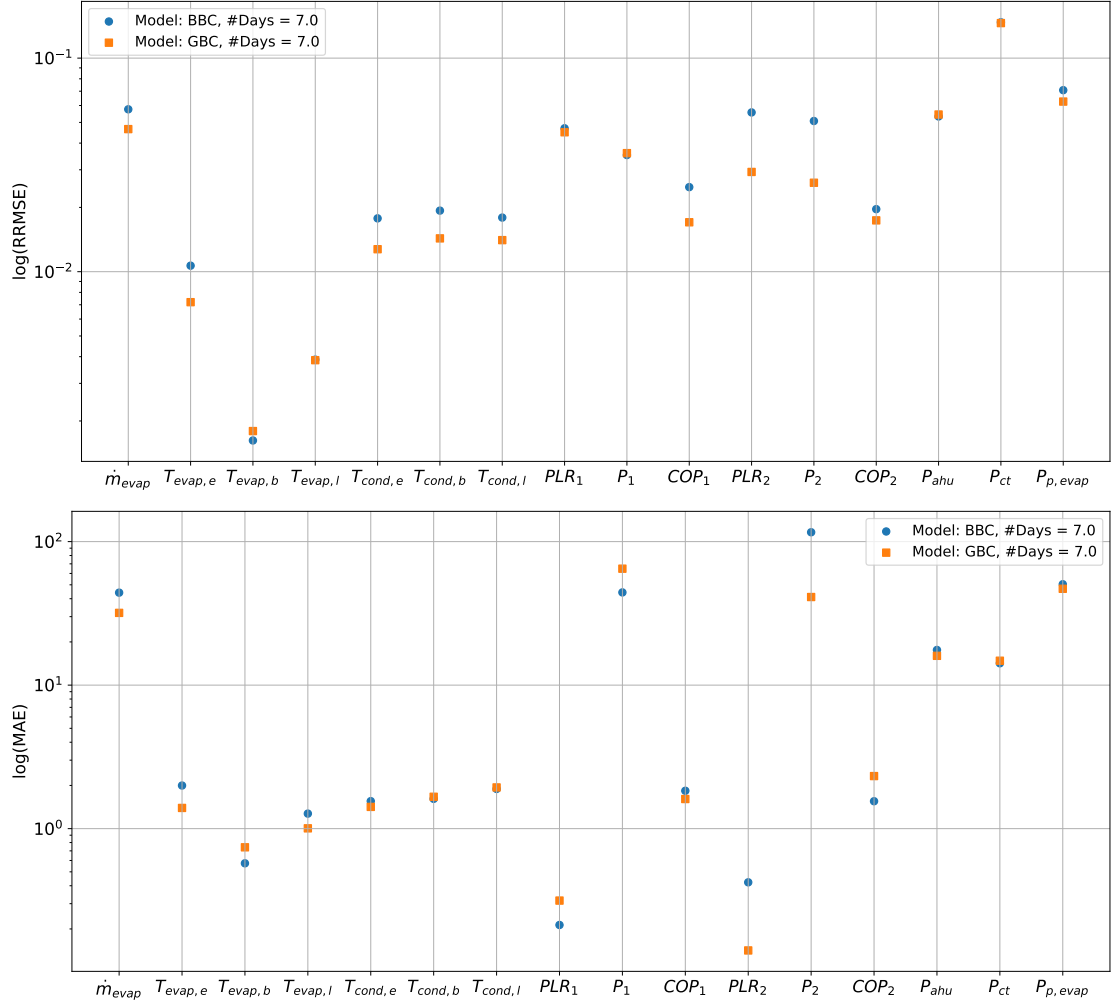


Figure 5.15: Comparison of Relative Root Mean Squared Error (top) and Maximum Absolute Error (bottom) across all predicted variables for the two modeling strategies: Black-Box Component-based (BBC) and Gray-Box Component-based (GBC). The results correspond to using data equivalent to seven days for training.

It can be observed in Figure 5.15 that the gray-box model demonstrates generally higher accuracy than the black-box model across most variables, despite both models yielding similar worst-case prediction errors. This, together with the results illustrated in Figure 5.14, suggests that the gray-box approach is more accurate overall, independently of the amount of training data. However, it should be noted that both modeling strategies may still exhibit instability when trained on limited datasets, emphasizing the importance of sufficient data for reliable model performance.

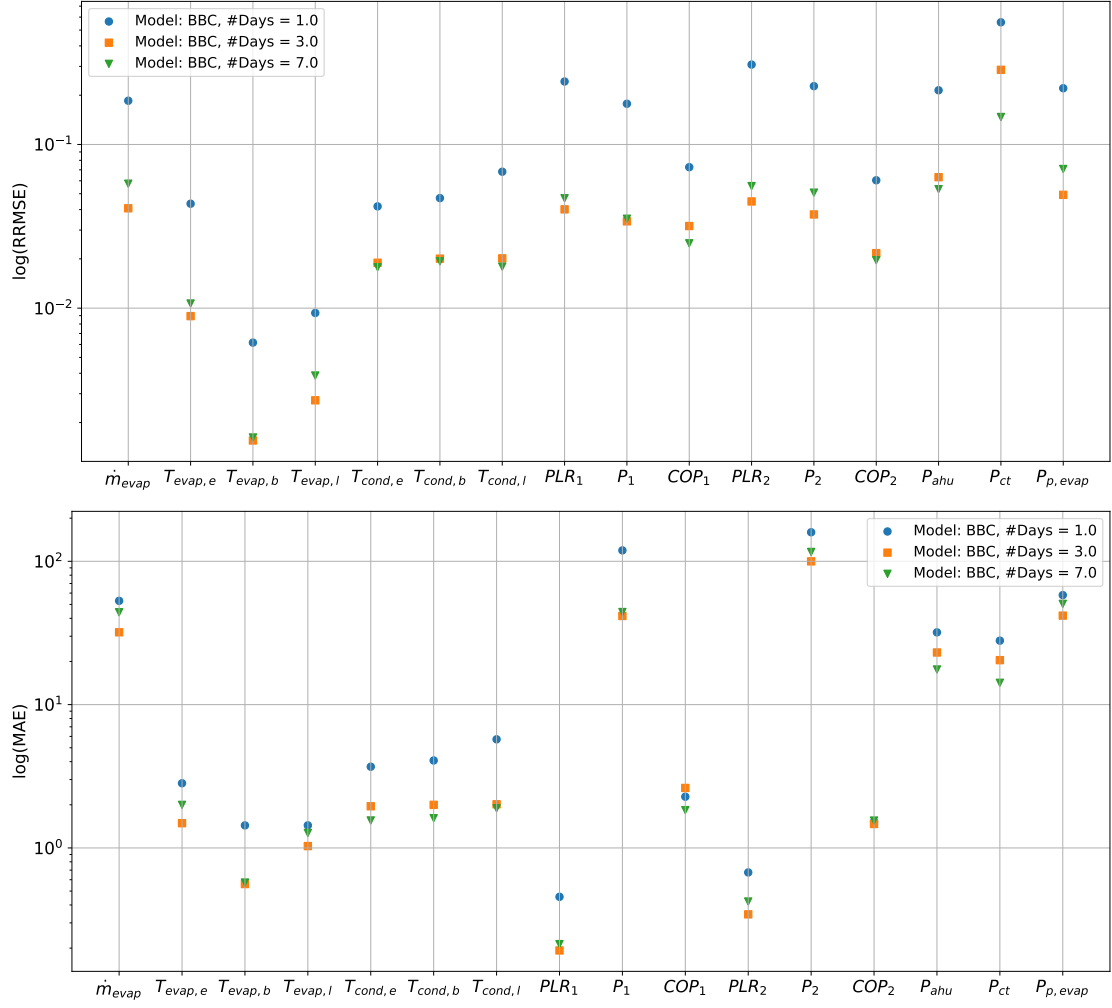


Figure 5.16: Comparison of Relative Root Mean Squared Error (top) and Maximum Absolute Error (bottom) across all predicted variables for the Black-Box Component-based (BBC) approach using different training dataset sizes.

From Figure 5.16 it is evident that increasing the size of the training dataset from one day to three days leads to a noticeable improvement in model performance, as reflected by the reduced RRMSE and maximum absolute error. However, further increasing the dataset to seven days does not yield significant additional benefits, since the error metrics for each variable remain very similar.

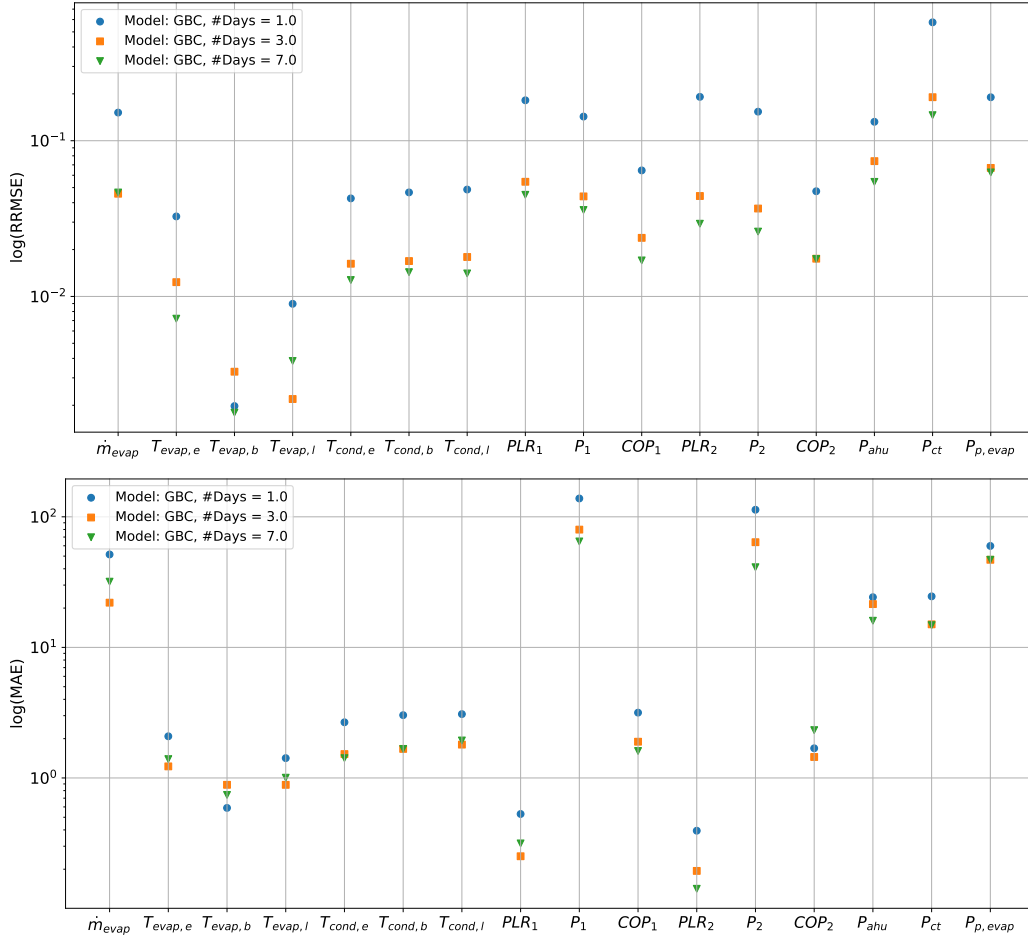


Figure 5.17: Comparison of Relative Root Mean Squared Error (top) and Maximum Absolute Error (bottom) across all predicted variables for the Gray-Box Component-based (GBC) approach using different training dataset sizes.

In contrast, [Figure 5.17](#) shows that there is a slight benefit in extending the training data from three days to a full week, particularly in terms of slightly reduced RRMSE values. While the improvements are modest, they suggest that generating a larger training dataset could be worthwhile if the goal is to maximize predictive performance.

Therefore, in this section it has been found that using physical knowledge of the system improves the performance of the model even when a lot of data is available. Furthermore, it has been observed that increasing the amount of training data may not improve the performance of the black-box model while the gray-box model predictive accuracy has increased.

5.2.3 Extrapolation Analysis

Another important challenge associated with neural networks is their reliance on training data that sufficiently spans the range of variability expected during deployment. Without adequate coverage of the input space, a model's ability to generalize to unseen conditions

can be significantly compromised. To address this, the present section investigates the impact of incorporating theoretical knowledge on the model's extrapolation capabilities.

The models considered in this analysis are the gray-box component-based and the black-box component-based approaches, using the network sizes that yielded the best performance in Subsection 5.2.1.

Figure 5.18 illustrates the different IT load profiles used to generate the datasets for this experiment. The models are trained and validated on datasets containing three days of data, with setpoints varying according to a random step function and with a constant or varying IT load. Evaluation is performed on a seven-day dataset in which the setpoint profile also follows a random step function; however, the IT load profile differs from that used during training.

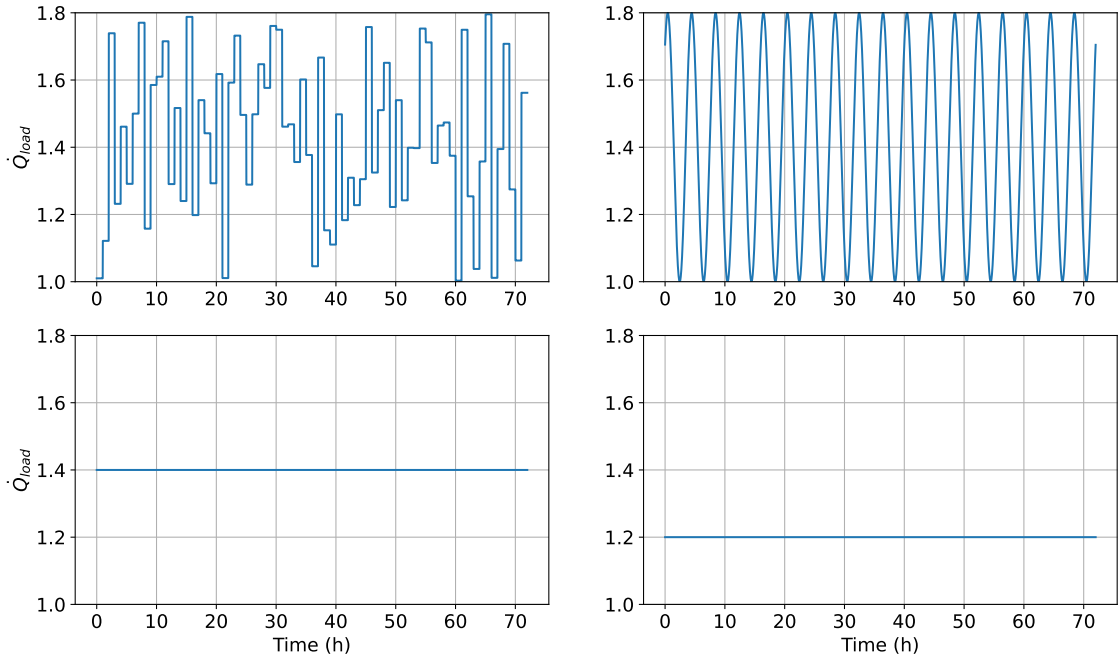


Figure 5.18: Time evolution over three days for different $\dot{Q}_{\text{load}}$ profiles: random step function every hour (top left), sinusoidal with periodicity of one hour (top right), constant at 1.4 MW (bottom left) and constant at 1.2 MW (bottom right).

It was observed that training both models on the $\dot{Q}_{\text{load}} = 1.4\text{MW}$ profile resulted in unstable models when applied to the varying IT load test datasets for predicting the evolution of temperatures and mass flow rate, as shown in Figure 5.19. Such instability was expected, as datasets with constant IT load do not contain the dynamic information necessary to model systems under varying $\dot{Q}_{\text{load}}$ conditions. However, the fact that the models also exhibit instability when tested on datasets with $\dot{Q}_{\text{load}} = 1.2\text{MW}$ suggests an additional issue. Specifically, the input and output variables may fall outside the distribution seen during training, indicating that the models are overfitting to the limited training data and fail to generalize when exposed to previously unseen operating conditions.

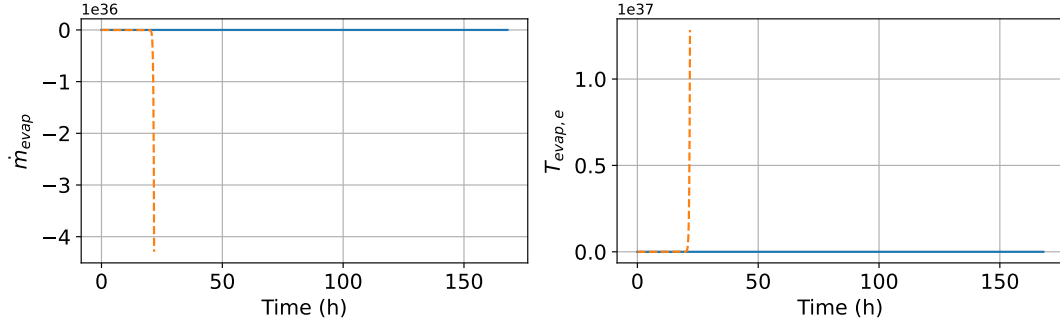


Figure 5.19: Trajectories of the mass flow rate and the temperature of the chilled water leaving the IT room using the test dataset and an unstable model. The predictions were produced by the Black-Box Component-based (BBC) model. In both plots, the true values are represented by a solid blue line while the model predictions are shown as dashed orange lines.

On the other hand, training on the dataset where the IT load varies according to a random step function occasionally led to unstable behavior for the black-box model, while the gray-box model remained stable under the same conditions. This suggests that incorporating physical knowledge into the modeling framework can improve robustness in extrapolation scenarios. Nevertheless, the results also highlight that a sufficiently rich and representative training dataset remains essential, even when physics-based components are included.

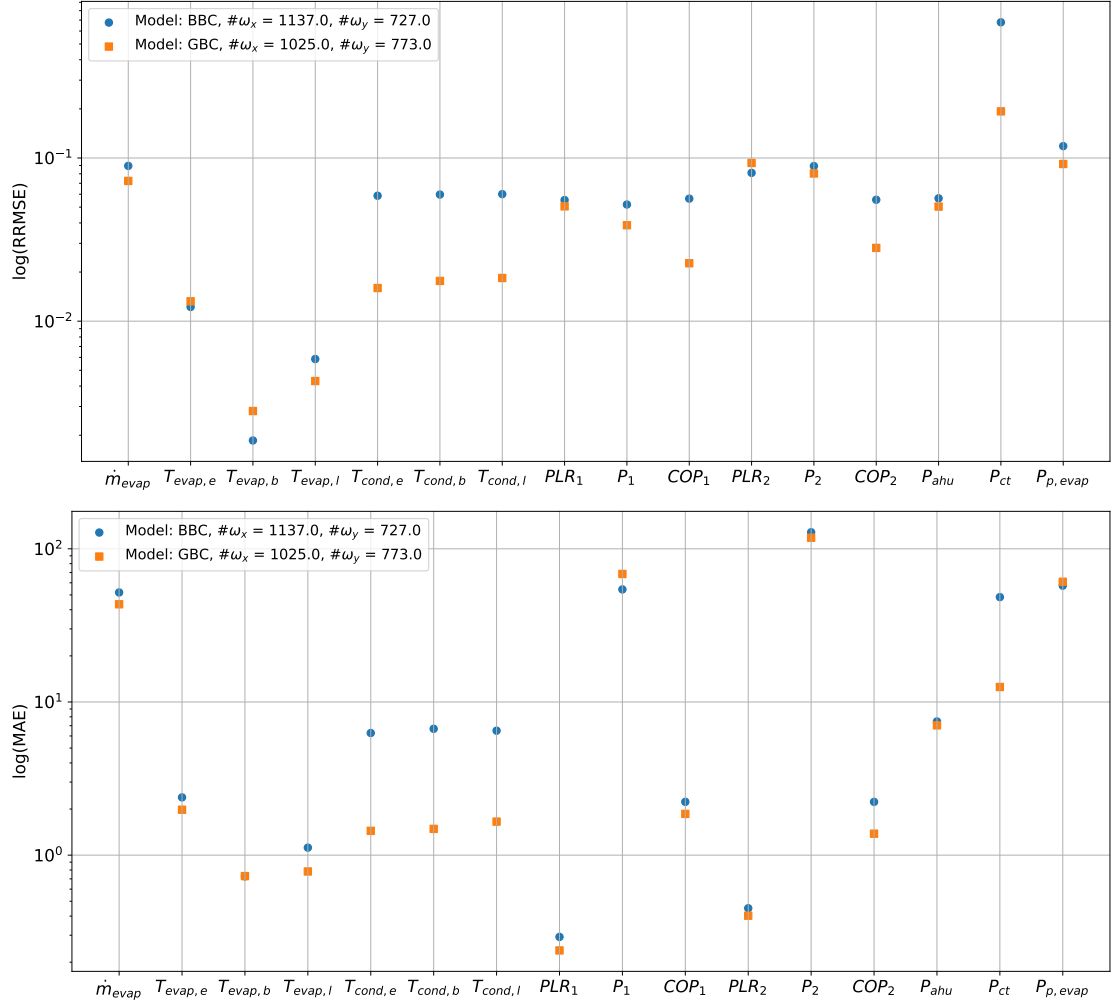


Figure 5.20: Comparison of Relative Root Mean Squared Error (top) and Maximum Absolute Error (bottom) across all predicted variables for the Black-box Component-Based and Gray-Box Component-based (GBC) approaches. The It load varied following a random step function in the training dataset and a sinusoidal function in the testing dataset.

Figure 5.20 illustrates that using the gray-box approach provides more accurate predictions than the black-box model when extrapolating to unseen data characterized by different IT load profiles. Therefore, incorporating physical knowledge into the modeling process can improve the model's performance by guiding the learning and maintaining stability even in extrapolation settings.



Figure 5.21: Comparison of Relative Root Mean Squared Error (top) and Maximum Absolute Error (bottom) across all predicted variables for the Black-box Component-Based and Gray-Box Component-based (GBC) approaches. The It load varied following a random step function in the training dataset and a constant function at 1.4 MW in the testing dataset.

In contrast, [Figure 5.20](#) shows that both modeling strategies achieve similar levels of accuracy when the training dataset includes a varying IT load and the testing dataset features a constant IT load. Moreover, both RRMSE and maximum absolute error are lower across all variables compared to the previous extrapolation scenario, with the improvement being particularly noticeable for the black-box model. This outcome is expected, as the training dataset captures all the relevant dynamics and value ranges present in the test dataset, thereby minimizing extrapolation effects.

In conclusion, gray-box models demonstrate a clear advantage over black-box models in extrapolation scenarios, achieving comparable or superior prediction accuracy. Moreover, they have been argued to exhibit greater robustness than their black-box counterparts.

5.2.4 Analysis of Training Data Variability

In previous experiments, the training datasets were designed to ensure sufficient excitation of the system by simultaneously varying three different setpoints. While this approach is valuable for evaluating the model's ability to learn system dynamics, it does not reflect the conditions typically encountered in real-world applications. In practice, companies are often reluctant to modify HVAC operating conditions purely for data collection purposes, due to safety and operational constraints. As a result, historical datasets used for model development are usually limited in variability and may only involve changes to a single control variable.

In this section, the models that are used correspond to the gray-box component-based and black-box component-based that performed best in [Subsection 4.2.2](#). Furthermore, we use datasets in which only the setpoint for the leaving water temperature of the first chiller ($T_{evap,b}^{set}$) is varied over time. The other two setpoints, $T_{evap,l}^{set}$ and $T_{cond,e}^{set}$, are held constant. This setup is therefore more similar to the operational data that would be available for HVAC systems.

The aim of this analysis is to assess whether the proposed modeling strategies can still learn meaningful system dynamics and make reliable predictions when trained on such limited datasets. Additionally, this experiment allows us to evaluate the extent to which physical priors incorporated into the gray-box model can compensate for the lack of excitation in the training data.

To conduct the analysis, the models are trained and validated on data equivalent to three days of operation with the setpoint profiles described before and with a varying IT load following a random step function. After training, the models are evaluated on a 7-day test dataset with similar setpoint and IT load profiles.

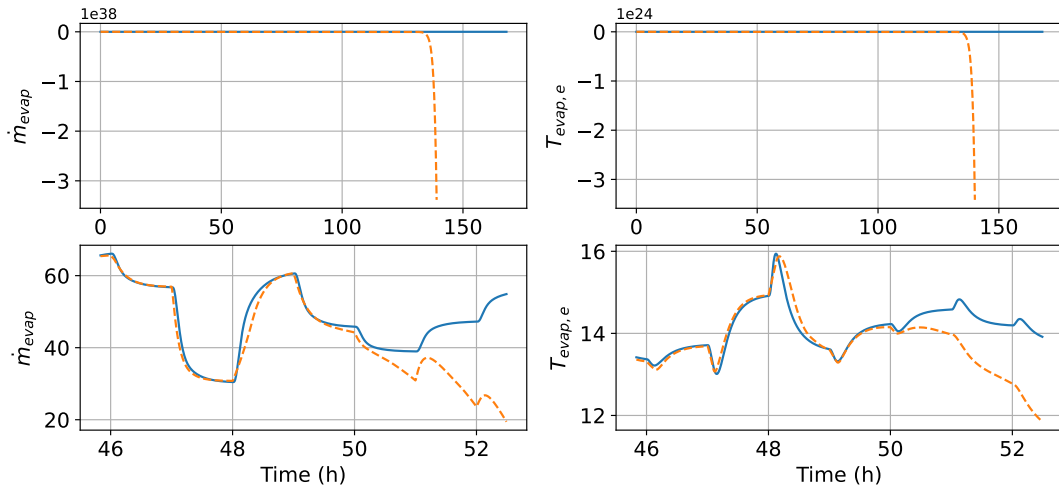


Figure 5.22: Trajectories of the mass flow rate and the temperature of the chilled water leaving the IT room using the test dataset and an unstable model. The predictions were produced by the Black-Box Component-based (BBC) model. In all plots, the true values are represented by a solid blue line while the model predictions are shown as dashed orange lines.

Figure 5.22 illustrates that the model was able to accurately predict the mass flow rate and the temperature of the chilled water after absorbing the heat from the IT room until a certain point in which the model was unstable. A similar behaviour was observed for the gray-box modeling, indicating that these models require a rich enough training dataset to not be unstable.

Chapter 6

Conclusions

This thesis has investigated data-driven and physics-informed modeling strategies for a chiller plant, with a particular focus on evaluating the potential benefits of combining physical knowledge with machine learning. Four modeling strategies were developed and compared: black-box and gray-box models, each implemented with and without a component-based architecture. All models were constructed using Neural Ordinary Differential Equations (NODEs), which enable the integration of physics-based differential equations with neural networks to capture unmodeled or complex dynamics.

As a first step, the suitability of Neural Ordinary Differential Equations (NODEs) for modeling HVAC systems was evaluated using a simplified setup comprising two isolated chillers. The training and validation datasets consisted of three days of data sampled at 30-second intervals, with varying setpoints applied to ensure adequate system excitation. The results demonstrated that the proposed methodology yields highly accurate temperature predictions, with Relative Root Mean Squared Error (RRMSE) values consistently below 1%. In contrast, the prediction of output variables exhibited higher errors, with RRMSE values remaining below 5%.

With respect to modeling strategies, adopting a component-based formulation reduced the number of trainable parameters required to achieve accuracy comparable to that of the non-component-based approach. A similar reduction in model complexity was observed when incorporating physical knowledge into the NODE formulation. However, introducing physics to model the relationships between system states to output variables did not yield a comparable reduction in parameter count for a given level of prediction accuracy.

The analysis was extended to the full chiller plant, which includes two chillers, pumps, cooling towers, and the IT room. All four modeling strategies were applied using 3 days of training data with varied setpoints and IT load. The results confirmed previous findings: component-based approaches required fewer parameters to achieve similar accuracy as their non-component-based counterparts. This trend was also observed when physics were introduced in the NODEs, while introducing physics to predict the outputs did not involve a reduction in the model complexity.

Overall, the models achieved high predictive accuracy for the time evolution of temperatures, with RRMSE values generally below 2%, demonstrating the modeling capabilities of NODEs. However, the worst-case temperature prediction errors exceeded 1°C, which may be unacceptable for HVAC systems requiring tight temperature control. Output

predictions remained more challenging, with RRMSE values around 6%. Both gray-box and black-box models were able to accurately capture slow transients and steady states.

The impact of training data availability was then studied by varying the size of the dataset. The gray-box and black-box component-based approaches were trained on 1, 3 and 7 days of data. The results showed that gray-box models were more robust and accurate when training data was limited, confirming their ability to alleviate data requirements. Interestingly, increasing the training duration to 7 days did not significantly improve black-box performance but did lead to modest gains for gray-box models.

Extrapolation capabilities were also assessed by evaluating model performance on test datasets with different IT load profiles than those seen during training. Models trained on constant IT load profiles failed when applied to test datasets with varying or lower IT loads. Even when trained on variable profiles, the black-box models struggled under extrapolation, often producing unstable outputs. In contrast, gray-box models demonstrated improved stability and better predictive performance, indicating that incorporating physics can mitigate extrapolation issues.

Finally, the models were tested on a more realistic scenario, where only one setpoint varied in the training data—reflecting the limited variability typical of historical datasets. In this case, both gray-box and black-box models exhibited instability when applied to unseen data, regardless of the amount of training data. These results highlight that, although the incorporation of physics enhances robustness and generalization, a minimum level of excitation in the training data remains necessary to ensure stable and accurate model behavior.

In summary, this thesis demonstrates that gray-box modeling provides a good balance between the flexibility of data-driven methods and the structural advantages of incorporating physical knowledge. Gray-box models achieve high accuracy with fewer parameters, exhibit greater robustness in extrapolation scenarios, and outperform purely data-driven models when training data is limited. However, the integration of physics does not eliminate the need for informative training data as sufficient system excitation remains essential to accurately capture dynamic behavior. These findings support the adoption of physics-informed modeling frameworks over purely data-driven approaches for HVAC applications. Nevertheless, further refinement is required to meet the strict performance demands of such systems, particularly in reducing the maximum absolute error to ensure reliable and precise operation.

Chapter 7

Future Work

Several directions for future research can be identified based on the findings and limitations of this project.

An immediate extension would be to improve the theory-based equations used in the gray-box models. For instance, the current formulation relies on first-order differential equations; incorporating second-order dynamics could enhance the accuracy of temperature predictions.

Additionally, refining the ALAMO model used to predict the power consumption of the chillers, or incorporating physics-based relationships into the modeling of other outputs, such as the power consumption of the cooling tower or the Air Handling Unit, may lead to improved predictive performance. This suggestion is supported by the observation in this project that embedding physical knowledge into the modeling process generally enhances accuracy.

Another promising direction involves incorporating discrete variables into the models, such as the on/off status of different components. This project focused exclusively on summer conditions, during which all components are typically active due to high outdoor temperatures. However, under different conditions, components may cycle off to reduce energy consumption. Accurately modeling such behavior would enable a more realistic and comprehensive representation of system dynamics.

Finally, another valuable extension of this work would be to model the time delays introduced by the piping system. In the models that have been created, the transport of water between components is assumed to occur instantaneously. However, in practice, these transfers involve delays due to the physical distance and flow rate of water in the pipes. Neglecting these delays can lead to temporal misalignments between control actions and their observed effects. Therefore, incorporating transport delays into the model would improve the accuracy of dynamic predictions and enhance the model's ability to replicate real system behavior.

Chapter 8

Appendix

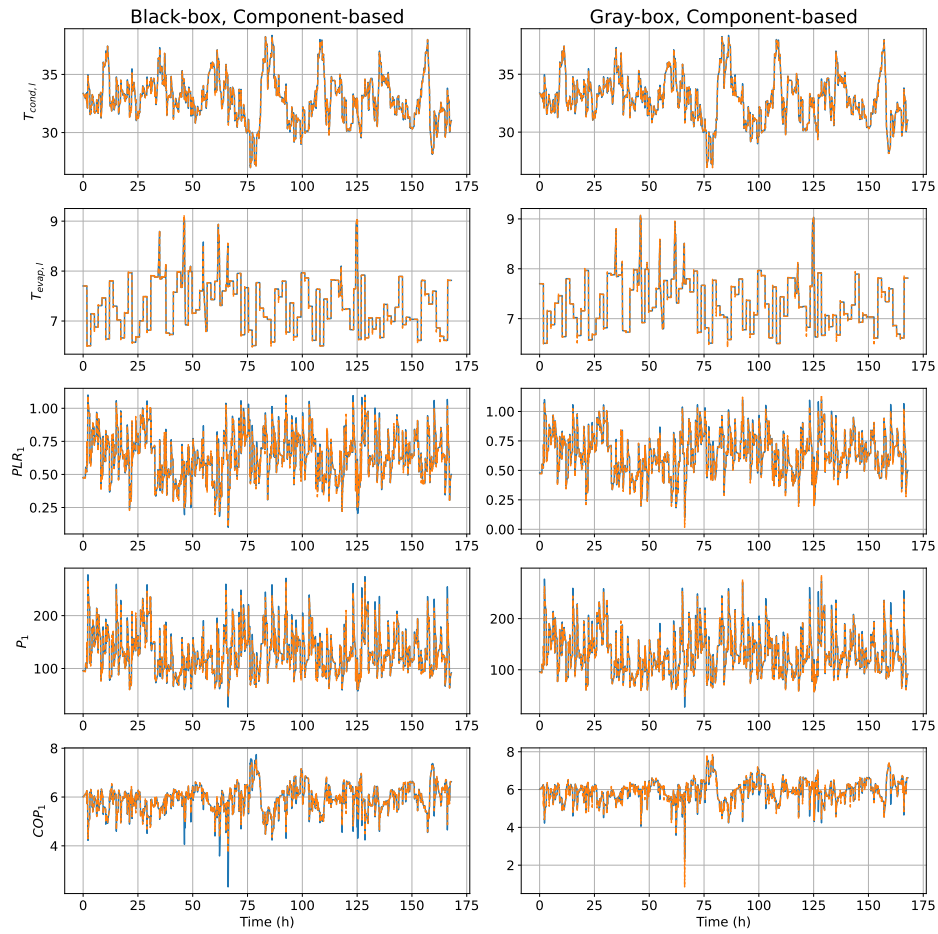


Figure 8.1: Comparison of the predicted trajectories for selected state and output variables using the test dataset. The left column shows the predictions produced by the Black-Box Component-based (BBC) model, while the right column shows those from the Gray-Box Component-based (GBC) model. In all plots, the true values are represented by a solid blue line, and the model predictions are shown as dashed orange lines.

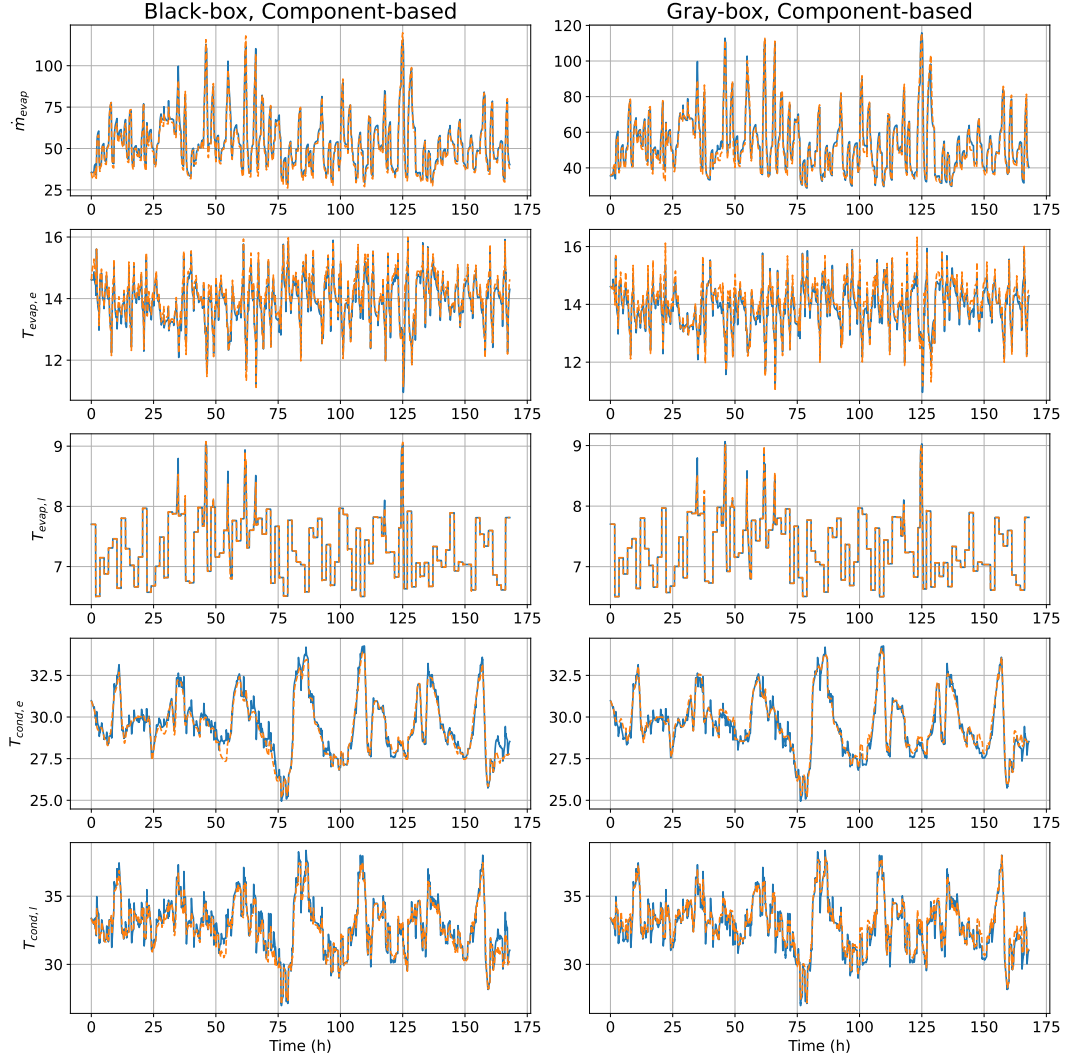


Figure 8.2: Comparison of the predicted trajectories for selected state variables using the test dataset. The left column shows the predictions produced by the Black-Box Component-based (BBC) model, while the right column shows those from the Gray-Box Component-based (GBC) model. In all plots, the true values are represented by a solid blue line, and the model predictions are shown as dashed orange lines.

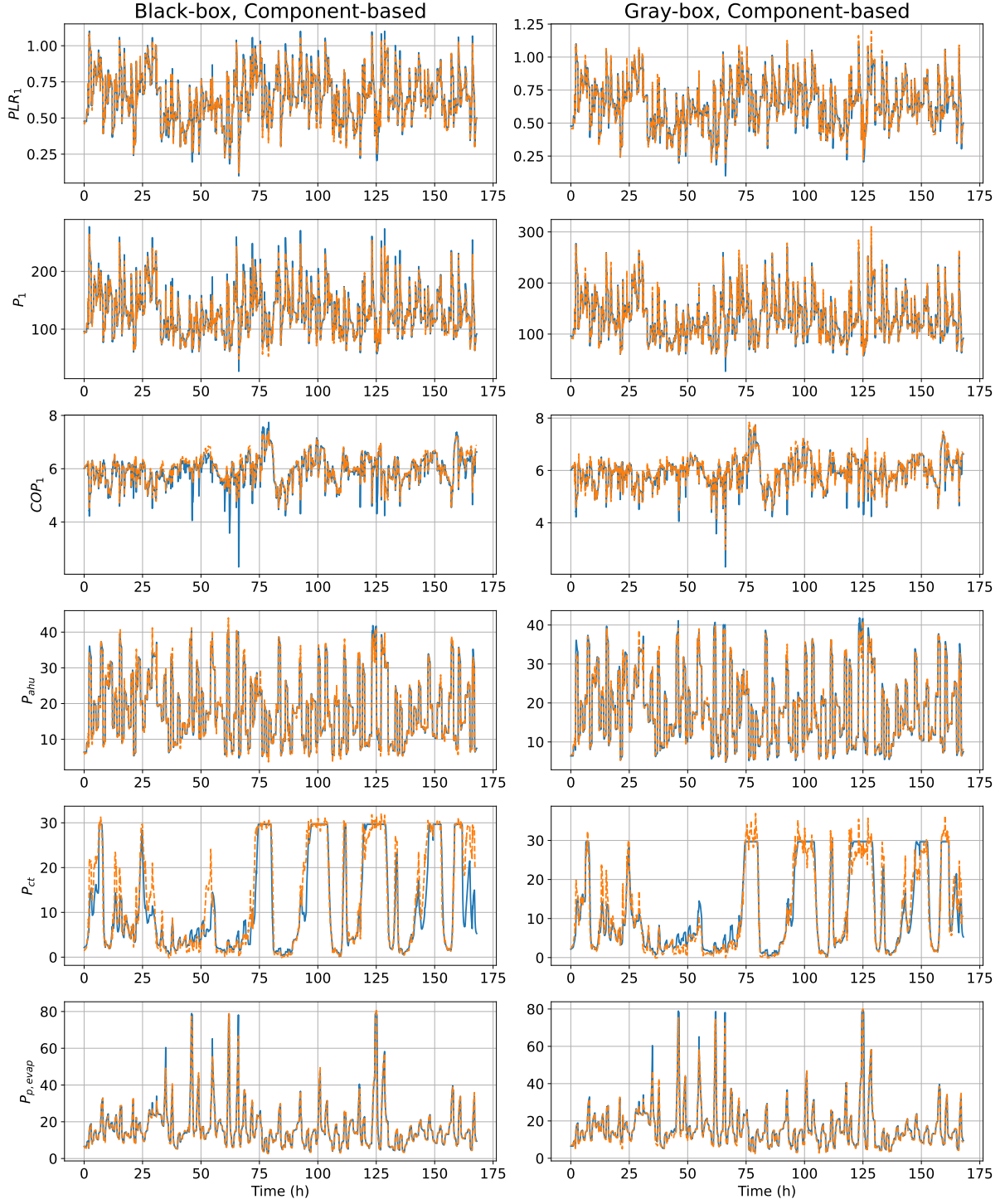


Figure 8.3: Comparison of the predicted trajectories for selected output variables using the test dataset. The left column shows the predictions produced by the Black-Box Component-based (BBC) model, while the right column shows those from the Gray-Box Component-based (GBC) model. In all plots, the true values are represented by a solid blue line while the model predictions are shown as dashed orange lines.

Chapter 9

Bibliography

- [1] Hannah Ritchie, Pablo Rosado, and Max Roser. “Energy Production and Consumption”. In: *Our World in Data* (2020). URL: <https://ourworldindata.org/energy-production-consumption>.
- [2] Global Alliance for Buildings and Construction (GlobalABC). *2021 Global Status Report for Buildings and Construction: Towards a Zero-emission, Efficient and Resilient Buildings and Construction Sector*. Tech. rep. United Nations Environment Programme, 2021. URL: <https://www.unep.org/resources/report/2021-global-status-report-buildings-and-construction>.
- [3] M. González-Torres et al. “A review on buildings energy information: Trends, end-uses, fuels and drivers”. In: *Energy Reports* 8 (2022), pp. 626–637. ISSN: 2352-4847. DOI: <https://doi.org/10.1016/j.egyr.2021.11.280>. URL: <https://www.sciencedirect.com/science/article/pii/S235248472101427X>.
- [4] Daisuke Murakami et al. *Climate-economy projections under shared socioeconomic pathways and net-zero scenarios*. 2025. arXiv: [2504.11721](https://arxiv.org/abs/2504.11721) [econ.GN]. URL: <https://arxiv.org/abs/2504.11721>.
- [5] International Energy Agency. *Renovation of near 20% of existing building stock to zero-carbon-ready by 2030 is ambitious but necessary*. Tech. rep. Paris: International Energy Agency, 2022. URL: <https://www.iea.org/reports/renovation-of-near-20-of-existing-building-stock-to-zero-carbon-ready-by-2030-is-ambitious-but-necessary>.
- [6] M. Fasiuddin and I. Budaiwi. “HVAC system strategies for energy conservation in commercial buildings in Saudi Arabia”. In: *Energy and Buildings* 43.12 (2011), pp. 3457–3466. ISSN: 0378-7788. DOI: <https://doi.org/10.1016/j.enbuild.2011.09.004>. URL: <https://www.sciencedirect.com/science/article/pii/S0378778811003926>.
- [7] Kevin Jay North. *Diagram showing the components of a liquid-cooled chiller*. <https://commons.wikimedia.org/w/index.php?curid=46265049>. Licensed under CC BY-SA 4.0. 2025.
- [8] Abdul Afram and Farrokh Janabi-Sharifi. “Review of modeling methods for HVAC systems”. In: *Applied Thermal Engineering* 67.1 (2014), pp. 507–519. ISSN: 1359-4311. DOI: <https://doi.org/10.1016/j.applthermaleng.2014.03.055>. URL: <https://www.sciencedirect.com/science/article/pii/S1359431114002348>.

- [9] Qiang Bi et al. “Advanced controller auto-tuning and its application in HVAC systems”. In: *Control Engineering Practice* 8.6 (2000), pp. 633–644. ISSN: 0967-0661. DOI: [https://doi.org/10.1016/S0967-0661\(99\)00198-7](https://doi.org/10.1016/S0967-0661(99)00198-7). URL: <https://www.sciencedirect.com/science/article/pii/S0967066199001987>.
- [10] Jakob Rehrl and Martin Horn. “Temperature control for HVAC systems based on exact linearization and model predictive control”. In: *2011 IEEE International Conference on Control Applications (CCA)*. 2011, pp. 1119–1124. DOI: [10.1109/CCA.2011.6044437](https://doi.org/10.1109/CCA.2011.6044437).
- [11] Gongsheng Huang. “Model predictive control of VAV zone thermal systems concerning bi-linearity and gain nonlinearity”. In: *Control Engineering Practice* 19.7 (2011), pp. 700–710. ISSN: 0967-0661. DOI: <https://doi.org/10.1016/j.conengprac.2011.03.005>. URL: <https://www.sciencedirect.com/science/article/pii/S0967066111000451>.
- [12] Gongsheng Huang, Shengwei Wang, and Xinhua Xu and. “Robust Model Predictive Control of VAV Air-Handling Units Concerning Uncertainties and Constraints”. In: *HVAC&R Research* 16.1 (2010), pp. 15–33. DOI: [10.1080/10789669.2010.10390890](https://doi.org/10.1080/10789669.2010.10390890). URL: <https://doi.org/10.1080/10789669.2010.10390890>.
- [13] Ding Lixing et al. “Support vector regression and ant colony optimization for HVAC cooling load prediction”. In: *2010 International Symposium on Computer, Communication, Control and Automation (3CA)*. Vol. 1. 2010, pp. 537–541. DOI: [10.1109/3CA.2010.5533861](https://doi.org/10.1109/3CA.2010.5533861).
- [14] Olivier Morisot and Dominique Marchio. “Fault detection and diagnosis on HVAC variable air volume system using artificial neural network”. In: *Proc. IBPSA Building Simulation* (1999). URL: https://scholar.google.com/scholar_lookup?title=Fault%20detection%20and%20diagnosis%20on%20HVAC%20variable%20air%20volume%20system%20using%20artificial%20neural%20network&author=O.%20Morisot&publication_year=1999&pages=1027-1034.
- [15] Andrew Kusiak, Mingyang Li, and Zijun Zhang. “A data-driven approach for steam load prediction in buildings”. In: *Applied Energy* 87.3 (2010), pp. 925–933. ISSN: 0306-2619. DOI: <https://doi.org/10.1016/j.apenergy.2009.09.004>. URL: <https://www.sciencedirect.com/science/article/pii/S0306261909003808>.
- [16] Chi-man Jacob Yiu et al. “Statistical modelling and forecasting schemes for air-conditioning system”. In: (2008). URL: <https://theses.lib.polyu.edu.hk/handle/200/2728>.
- [17] Guang-Yu Jin et al. “A hybrid water-cooled centrifugal chiller model”. In: *2011 6th IEEE Conference on Industrial Electronics and Applications*. 2011, pp. 2298–2303. DOI: [10.1109/ICIEA.2011.5975975](https://doi.org/10.1109/ICIEA.2011.5975975).
- [18] A.P. Wemhoff and M.V. Frank. “Predictions of energy savings in HVAC systems by lumped models”. In: *Energy and Buildings* 42.10 (2010), pp. 1807–1814. ISSN: 0378-7788. DOI: <https://doi.org/10.1016/j.enbuild.2010.05.017>. URL: <https://www.sciencedirect.com/science/article/pii/S0378778810001799>.

-
- [19] M. Mossolli, K. Ghali, and N. Ghaddar. “Optimal control strategy for a multi-zone air conditioning system using a genetic algorithm”. In: *Energy* 34.1 (2009), pp. 58–66. ISSN: 0360-5442. DOI: <https://doi.org/10.1016/j.energy.2008.10.001>. URL: <https://www.sciencedirect.com/science/article/pii/S0360544208002570>.
- [20] Guang-Yu Jin et al. “A simplified modeling of mechanical cooling tower for control and optimization of HVAC systems”. In: *Energy Conversion and Management* 48.2 (2007), pp. 355–365. ISSN: 0196-8904. DOI: <https://doi.org/10.1016/j.enconman.2006.07.010>. URL: <https://www.sciencedirect.com/science/article/pii/S0196890406002317>.
- [21] Xiuying Yan, Qingchang Ren, and Qinglong Meng. “Global optimization of VAV air conditioning system”. In: *2010 8th World Congress on Intelligent Control and Automation*. 2010, pp. 5077–5081. DOI: [10.1109/WCICA.2010.5554517](https://doi.org/10.1109/WCICA.2010.5554517).
- [22] Ping Cui et al. “Simulation of hybrid ground-coupled heat pump with domestic hot water heating systems using HVACSIM+”. In: *Energy and Buildings* 40.9 (2008), pp. 1731–1736. ISSN: 0378-7788. DOI: <https://doi.org/10.1016/j.enbuild.2008.03.001>. URL: <https://www.sciencedirect.com/science/article/pii/S0378778808000625>.
- [23] Jiankui Li and Zhidong Pan. “Annihilator-preserving maps, multipliers, and derivations”. In: *Linear Algebra and its Applications* 432.1 (2010), pp. 5–13. ISSN: 0024-3795. DOI: <https://doi.org/10.1016/j.laa.2009.06.002>. URL: <https://www.sciencedirect.com/science/article/pii/S0024379509002961>.
- [24] Jonathan Leclerc, Frederic Wurtz, and Etienne Wurtz. “Low order envelope model for optimised predictive control of indoor temperature: Development methodology and calibration with a numerical model”. In: 2013, 2341 – 2348. URL: <https://www.scopus.com/inward/record.uri?eid=2-s2.0-84886704783&partnerID=40&md5=134be1256a5e12c67fe7cfeba0ebd17f>.
- [25] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. ISBN: 9780262035613. URL: <http://www.deeplearningbook.org>.
- [26] Diederik P. Kingma and Jimmy Ba. “Adam: A Method for Stochastic Optimization”. In: *CoRR* abs/1412.6980 (2014). URL: <https://api.semanticscholar.org/CorpusID:6628106>.
- [27] Sébastien Bubeck. *Convex Optimization: Algorithms and Complexity*. 2015. arXiv: [1405.4980](https://arxiv.org/abs/1405.4980) [math.OA]. URL: <https://arxiv.org/abs/1405.4980>.
- [28] Atilim Gunes Baydin et al. *Automatic differentiation in machine learning: a survey*. 2018. arXiv: [1502.05767](https://arxiv.org/abs/1502.05767) [cs.LG]. URL: <https://arxiv.org/abs/1502.05767>.
- [29] PyTorch Developers. *Autograd Mechanics*. <https://docs.pytorch.org/docs/stable/notes/autograd.html>. PyTorch. 2024.
- [30] Kaiming He et al. “Deep Residual Learning for Image Recognition”. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 770–778. DOI: [10.1109/CVPR.2016.90](https://doi.org/10.1109/CVPR.2016.90).
- [31] Ricky T. Q. Chen et al. *Neural Ordinary Differential Equations*. 2019. arXiv: [1806.07366](https://arxiv.org/abs/1806.07366) [cs.LG]. URL: <https://arxiv.org/abs/1806.07366>.

- [32] J.C Butcher. “Differential and Difference Equations”. In: *Numerical Methods for Ordinary Differential Equations*. John Wiley & Sons, Ltd, 2016. Chap. 1, pp. 1–53. ISBN: 9781119121534. DOI: <https://doi.org/10.1002/9781119121534.ch1>. URL: <https://onlinelibrary.wiley.com/doi/abs/10.1002/9781119121534.ch1>.
- [33] Patrick Kidger. *On Neural Differential Equations*. 2022. arXiv: [2202.02435](https://arxiv.org/abs/2202.02435) [cs.LG]. URL: <https://arxiv.org/abs/2202.02435>.
- [34] Derek Onken and Lars Ruthotto. *Discretize-Optimize vs. Optimize-Discretize for Time-Series Regression and Continuous Normalizing Flows*. 2020. arXiv: [2005.13420](https://arxiv.org/abs/2005.13420) [cs.LG]. URL: <https://arxiv.org/abs/2005.13420>.
- [35] Sam McCallum and James Foster. *Efficient, Accurate and Stable Gradients for Neural ODEs*. 2025. arXiv: [2410.11648](https://arxiv.org/abs/2410.11648) [cs.LG]. URL: <https://arxiv.org/abs/2410.11648>.
- [36] “Modelica”. In: (2025). URL: <https://modelica.org/language/>.
- [37] Jan Drgona et al. “NeuroMANCER: Neural Modules with Adaptive Nonlinear Constraints and Efficient Regularizations”. In: (2023). URL: <https://github.com/pnnl/neuromancer>.
- [38] Nick Sahinidis. “The ALAMO approach to machine learning”. In: *26th European Symposium on Computer Aided Process Engineering*. Ed. by Zdravko Kravanja and Miloš Bogataj. Vol. 38. Computer Aided Chemical Engineering. Elsevier, 2016, p. 2410. DOI: <https://doi.org/10.1016/B978-0-444-63428-3.50410-0>. URL: <https://www.sciencedirect.com/science/article/pii/B9780444634283504100>.
- [39] *National Weather Service*. U.S. Department of Commerce. 2025. URL: <https://www.weather.gov/>.

Master's Theses in Mathematical Sciences 2025:E49
ISSN 1404-6342
LUNFBV-3002-2025
Computational Science
Centre for Mathematical Sciences
Lund University
Box 118, SE-221 00 Lund, Sweden
<http://www.maths.lu.se/>