A Comparative Study of Physics-Informed Machine Learning Methods for Modeling HVAC Systems

Tung L. Nguyen [1] and Truong Nghiem [1]

[1]Affiliation not available

October 30, 2023

Abstract

Machine learning (ML) methods have been used to model complex dynamical systems, such as heating, ventilation, and air conditioning (HVAC) systems, to overcome the difficulty and high cost of modeling these systems using physical principles. However, ML-based methods often require large amounts of data, have poor generalization performance, and lack physical consistency. Physics-informed machine learning (PIML) has recently been introduced to overcome these drawbacks by incorporating physical laws into learning. There is, however, an unmet need to evaluate commonly used PIML methods to demonstrate their benefits and compare their performance in practical applications. In this comparative study, we evaluated various PIML methods and physical properties for modeling HVAC systems using real data. We considered physics-informed neural network methods and constrained Gaussian process methods, as well as physical properties that can be easily obtained in practice, such as smoothness, boundedness, and monotonicity. Our results showed the substantial benefits of PIML in improving model accuracy and data efficiency, and allowed us to compare the different PIML methods and physical properties to provide meaningful conclusions and recommendations for applying PIML in practice.

A Comparative Study of Physics-Informed Machine Learning Methods for Modeling HVAC Systems

Tung L. Nguyen and Truong X. Nghiem
School of Informatics, Computing, and Cyber Systems
Northern Arizona University
tln229@nau.edu, truong.nghiem@nau.edu

***Abstract*— Machine learning (ML) methods have been used to model complex dynamical systems, such as heating, ventilation, and air conditioning (HVAC) systems, to overcome the difficulty and high cost of modeling these systems using physical principles. However, ML-based methods often require large amounts of data, have poor generalization performance, and lack physical consistency. Physics-informed machine learning (PIML) has recently been introduced to overcome these drawbacks by incorporating physical laws into learning. There is, however, an unmet need to evaluate commonly used PIML methods to demonstrate their benefits and compare their performance in practical applications. In this comparative study, we evaluated various PIML methods and physical properties for modeling HVAC systems using real data. We considered physics-informed neural network methods and constrained Gaussian process methods, as well as physical properties that can be easily obtained in practice, such as smoothness, boundedness, and monotonicity. Our results showed the substantial benefits of PIML in improving model accuracy and data efficiency, and allowed us to compare the different PIML methods and physical properties to provide meaningful conclusions and recommendations for applying PIML in practice.**

I. Introduction

Modeling a complex system, such as a heating, ventilation, and air conditioning (HVAC) system, using physical principles, also known as mechanistic modeling or first-principle modeling, requires extraordinary effort, time, and domain knowledge [1]. Modeling methods based on machine learning (ML) have been developed to overcome these challenges for buildings and other complex systems [2]. However, while having found great success in certain applications, purely data-driven modeling methods often require large amounts of data, have poor generalization performance and physical consistency, and have difficulty handling low quality data, due to their lack of physical insights [3]. To address these limitations, physics-informed machine learning (PIML) methods have recently emerged to incorporate known physical laws governing a system into the learning process [3]. It is an important step towards interpretable, robust, accurate, and physically consistent ML models of physical systems, by merging ML with physics-based modeling. PIML models are useful for prediction, simulation, control, and optimization, with often higher confidence of their performance and safety compared to non-physics-informed ML models.

This material is based upon work supported by the National Science Foundation under Grant No. 2138388 and Grant No. 2238296.

Several PIML approaches have been developed in recent years for different combinations of physical properties and ML models in various applications, such as constrained neural networks [4]–[6], lattice networks [7], and constrained Gaussian processes (GPs) [8, 9]. While ML methods have long been applied to HVAC system modeling and control, using e.g., conventional neural networks (NNs) [10, 11], regression trees and random forests [12, 13], and GPs [14, 15], PIML methods have only recently been applied to HVAC systems, mainly using NNs [4, 16]. Previous studies have compared a wide range of conventional ML methods, from simple nonlinear regressions to NNs, for HVAC applications [2, 17]. However, to our knowledge, no study has compared different PIML methods for modeling HVAC systems to yield insightful comparisons in terms of accuracy, data efficiency, and computation. Such a study can demonstrate the benefits of PIML methods for practical applications and provide guidance on selecting appropriate PIML methods and physical constraints to incorporate into models.

This paper presents a comparative study that evaluated and compared several PIML methods, ranging from NNs to GPs, for modeling HVAC systems using real data, which are representative of complex dynamical systems that can significantly benefit from data-driven modeling methods. Our study aimed to provide valuable insights into the use of PIML methods to improve the accuracy and data efficiency of learning in practical applications. Our main contributions are:

1) We evaluated and compared various PIML methods, including popular physics-informed neural network methods and constrained Gaussian process methods from the research literature.
2) We examined several physical properties that can be easily obtained in practice, e.g., smoothness, boundedness, and monotonicity, for integration into ML.
3) We used data from real HVAC systems for the study.
4) Our study rigorously compared the considered PIML methods and physical properties, resulting in insightful conclusions and practical recommendations. Further details can be found in Sec. III-E.

The remainder of the paper is organized as follows. Sec. II briefly reviews the physical properties and PIML methods investigated in our study. Sec. III presents the experiments,

results, and a comprehensive discussion in subsection III-E. The paper is concluded by Sec. IV.

II. Physics-Informed Machine Learning Methods for HVAC Systems

As mentioned earlier, PIML refers to methods for seamlessly integrating known physical properties of a system, e.g., conservation of energy and mass, monotonicity, and boundedness, into ML-based modeling of the system for enhancing the model's fidelity, data efficiency, interpretability, robustness, and reliability [3]. This section reviews the PIML methods and the physical properties relevant to HVAC systems that were examined in our study.

A HVAC system is a mechanical dynamical system that must adhere to the laws of physics. Our study considered a variable-air-volume (VAV) HVAC system at the zone level, illustrated in Fig. 3, which includes a room, a thermostat for regulating the room temperature, and a reheat coil for reheating the supply air into the room. The system and its variables will be detailed in Sec. III. While many physical properties can be inferred from the first principles applicable to the system, we focused on intuitive properties that can be obtained easily from a high-level understanding of the operation and physics of the system. In particular, three properties were considered for integration into ML: smoothness, boundedness, and monotonicity. In the following, $f : \mathbb{R}^n \to \mathbb{R}$ denotes the function of a model that takes an input vector $x \in \mathbb{R}^n$ to produce a scalar output $f(x) \in \mathbb{R}$, and $x^{(d)}$ denotes the d-th element of vector x.

- **Smoothness:** This property refers to the fact that the physical variables of the system, e.g., the room temperature or the reheated supply air temperature, do not change abruptly. This means that the absolute change in the outputs at two consecutive steps in a time series is fairly small, i.e., $|f(x_{i+1}) - f(x_i)|$ is small (or, in some cases, limited by a known bound).
- **Boundedness:** The output $f(x)$ has known upper-bound and/or lower-bound. For instance, the damper position and the reheat valve position must be between 0% and 100%, or the mass flow rate of the supply air has known minimum value (due to ventilation requirement) and maximum value (due to mechanical limit).
- **Monotonicity:** The function f is increasing (or decreasing) monotonic w.r.t. input dimension $d \in \{1, \dots, n\}$ if for any two input vectors x_1 and x_2 such that $x_1^{(d)} \leq x_2^{(d)}$ and $x_1^{(i)} = x_2^{(i)}$ for all $i \neq d$ we have $f(x_1) \leq f(x_2)$ (or $f(x_1) \geq f(x_2)$). For instance, the room temperature is increasing monotonic w.r.t. the supply air temperature since increasing the supply air temperature will increase the room temperature.

Using these properties for constraining a ML model can help it avoid behaviors that violate physics, improve its accuracy, and increase its data efficiency and robustness. Other properties have been examined in the literature, such as stability [4, 5]; however, we do not consider them in this work.

The rest of this section will review several PIML methods for integrating the above properties into NNs and GPs. It is assumed that a time-series dataset is given and is denoted by $\mathcal{D} = \{x_i, y_i\}_{i=1}^N$, which consists of inputs or features $x_i \in \mathbb{R}^n$ and the corresponding outputs or observations $y_i \in \mathbb{R}$. Our goal is to learn, from the dataset $\mathcal{D}$, a model $y = f(x)$ of the underlying process of $\mathcal{D}$, using different ML models. Furthermore, we will describe various methods for incorporating each physical property into the model.

A. Physics-Informed Neural Networks

NNs are a class of ML models inspired by the biological structure of the human brain. There have been numerous excellent references that detail NNs, such as [18]. We will use $\mathcal{N}_\theta(x)$ to denote a NN with input x and network parameters θ. Generally, methods for incorporating physics directly into NNs, yielding physics-informed neural networks (PINNs), can be categorized into (1) methods that tailor the NN architecture to guarantee implicit satisfaction of given physical laws (i.e., using inductive biases) and (2) methods that use appropriate loss functions and constraints for model training to explicitly enforce or favor certain physical laws in the resulting model (i.e., using learning biases) [3]. We will now summarize several methods belonging to these two classes for integrating smoothness, boundedness, and monotonicity into NNs.

1) Smoothness: Promoting smoothness in a NN model $\mathcal{N}_\theta$ can be achieved by adding a penalty term to the loss function that penalizes the output change, resulting in a loss-based method [4, 5]. This method directly follows from the earlier definition of smoothness. Specifically, the ordinary mean squared error (MSE) loss is augmented with a smoothness penalty term as $\mathcal{L}_{\text{MSE}}(\mathcal{N}_\theta; \mathcal{D}) = \frac{1}{N}\sum_{i=1}^{N} \|\mathcal{N}_\theta(x_i) - y_i\|^2 + \alpha \frac{1}{N-1}\sum_{i=1}^{N-1} \|\mathcal{N}_\theta(x_{i+1}) - \mathcal{N}_\theta(x_i)\|^2$, where α is the (typically small) penalty weight.

2) Boundedness: If the function f has known upper-bound u and/or lower-bound l, i.e., $l \leq f(x) \leq u$ for all x, its boundedness can be incorporated using both loss-based and architecture-based methods. The first method penalizes the model for violating the bounds by adding a penalty term for each bound as $\mathcal{L}_{\text{MSE}}(\mathcal{N}_\theta; \mathcal{D}) = \frac{1}{N}\sum_{i=1}^{N} \|\mathcal{N}_\theta(x_i) - y_i\|^2 + \alpha \frac{1}{N}\sum_{i=1}^{N} (\text{ReLU}(\mathcal{N}_\theta(x_i) - u) + \text{ReLU}(l - \mathcal{N}_\theta(x_i)))$, where $\text{ReLU}(z) = \max(0, z)$ is the standard ReLU function and α is the penalty weight. We will refer to this method as "soft-bound NN." The second method enforces the bounds on the output directly in the model architecture by using the following activation function for the output layer to saturate out-of-bounds output values: $f_{\text{out}}(z) = \text{ReLU}(u - l - \text{ReLU}(u - z)) + l$. We will refer to this method as "hard-bound NN."

3) Monotonicity: Several methods have been proposed in the literature for enforcing a monotonic NN. A minimax network uses a special architecture that combines the min and max operators in the output layer, illustrated in Fig. 1, and also constrains certain weights in the network to be positive or negative during training [6]. Another method uses a lattice network [7], which has three types of layers: a linear embedding layer, a calibration layer, and a lattice layer. The architecture of a lattice network is depicted in Fig. 2. A linear embedding layer is a function $\text{li}(x) = x^\top W$ where

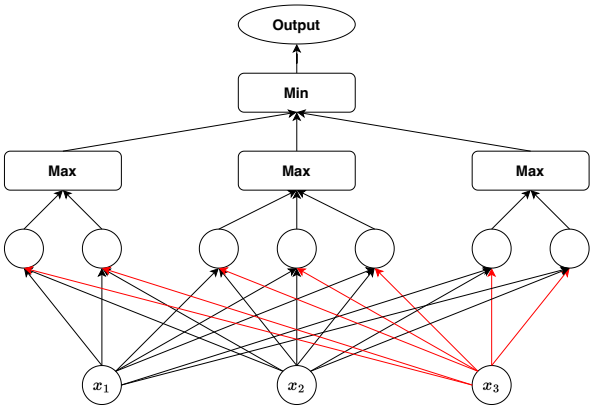

Fig. 1. Example of a minimax monotonic neural network. The input layer (bottom) has 3 inputs and the network is monotonic w.r.t. input x_3. The corresponding weights of x_3 are constrained to be positive if the network is increasing and negative if the network is decreasing w.r.t x_3.

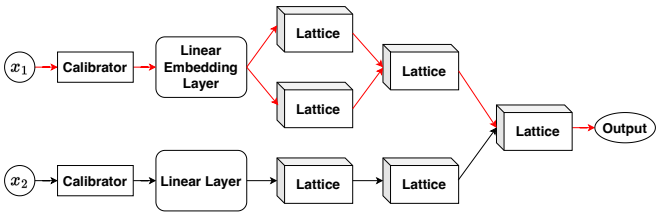

Fig. 2. Example of a monotonic lattice network w.r.t. input x_1. All the weights related to x_1 are positive if the network is increasing w.r.t x_1.

$x \in \mathbb{R}^m$ is the input and $W \in \mathbb{R}^{m \times n}$ are the parameters. If the network is increasing w.r.t. to x then $W_{i,j} \geq 0 \ \forall i, j$; conversely for decreasing monotonicity. A calibration layer is a piece-wise linear function $c(x) : \mathbb{R} \to \mathbb{R}$ for calibrating and normalizing an input feature. Finally, a lattice layer is a function $\text{la}(x) : \mathbb{R}^n \to \mathbb{R}$, which has the same mechanism as a calibration layer but in a higher dimension. More details can be found in [7].

B. Physics-Informed Gaussian Processes

Gaussian process regression (GPR) is a non-parametric, Bayesian approach to regression in ML. Readers are referred to [19] for a comprehensive treatment of GPs.

The kernel of a GP implicitly determines the smoothness of the learned function. For example, using the squared exponential (SE) kernel will result in functions that are smooth across many lengthscales [19]. Therefore, the smoothness property is incorporated into a GP by choosing an appropriate kernel.

Integrating boundedness and monotonicity into a GP can be achieved by linearly constrained GPs, which model processes that are constrained by certain linear inequalities. There are two prominent methods for constrained GPs: the authors of [8] developed finite-dimensional Gaussian approximation with linear inequality constraints, and Gaussian processes with linear operator inequality constraints were proposed in [9]. These methods are briefly described below.

1) Finite-Dimensional Gaussian Approximation with Linear Inequality Constraints [8]: This method approximates the target function by a linear combination of certain basic functions, e.g., $f(x) \approx \sum_{i=1}^{n} f(t_i)\phi_i(x)$ where $\{t_i\}_{i=1}^n$ is a set of pre-designed equidistant knots, i.e., $t_{i+1} - t_i = \Delta m \ \forall i$, and $\{\phi_i(x)\}_{i=1}^n$ is a set of *hat* functions, defined as

$$\phi_i(x) = \begin{cases} 1 - |\frac{x - t_i}{\Delta m}| & \text{if } |\frac{x-t_i}{\Delta m}| \leq 1 \\ 0 & \text{otherwise.} \end{cases}$$

The values at the knots $\zeta_i = f(t_i)$ are parameters of the model, with an assumed prior $\zeta \sim \mathcal{N}(0, \Gamma)$ where Γ is a known covariance matrix. Given the training dataset $\mathcal{D} = \{x_i, y_i\}_{i=1}^N = (X, Y)$, the parameters $\zeta = [\zeta_1, \ldots, \zeta_n]^\top$ are estimated to fit the approximate function $\hat{f}(x) = \sum_{i=1}^{n} \zeta_i \phi_i(x)$ to the data, that is $Y \sim \Phi\zeta$. To enforce boundedness and/or monotonicity of the learned model, appropriate constraints are imposed on the parameter vector ζ, e.g., $l \leq \zeta_i \leq u \ \forall i = 1, \ldots, n$ for boundedness and $\zeta_i \leq \zeta_{i+1} \ \forall i = 1, \ldots, n-1$ for increasing monotonicity. The posterior distribution of ζ is then estimated subject to the above constraints, by using truncated multivariate Gaussian sampling [20], then the distribution of the approximate function $\hat{f}(\cdot)$ is obtained. We call this method "approx-PIGP" for approximate physics-informed GP.

2) Gaussian Processes with Linear Operator Inequality Constraints [9]: This method enforces boundedness or monotonicity constraints on a set of pre-selected *inducing points* or *virtual observation locations*, $X^v = \{x_i^v\}_{i=1}^n$, during learning. In particular, to incorporate increasing monotonicity into the model, we can constrain $f'(x) > 0 \ \forall x \in X^v$. To bound the model output in the closed interval $[l, u]$, we can enforce $l \leq f(x) \leq u \ \forall x \in X^v$. This approach does not guarantee constraint satisfaction on the entire domain but only at the finite set of locations X^v. However, if X^v is dense enough, the constraints will hold everywhere with high probability. The learning goal is to find the distribution of $f|X, Y, X^v, C(X^v)$, where $C(X^v)$ represents the event that the constraint is satisfied for all points in X^v. Various methods can be used to sample the distribution of $C(X^v)$, such as the minimax tilting sampling method used in [9] or the probit likelihood used in [21]. We call this method "inducing-PIGP".

III. Experiments and Discussion

This section details the experiments of our study, discusses the results, and compares the considered PIML methods.

A. System and Data

Our study used actual data from the HVAC system of the building of the School of Informatics, Computing, and Cyber Systems at Northern Arizona University. Data from two different rooms was collected during the heating season in November 2022 for the study. The first room, room 102, is a classroom in the middle of the building's first floor, with no windows nor walls to the outside of the building, hence the outside weather has no direct impact on the room's environment. The second room, room 106, is a laboratory at a corner of the building's first floor, with two large adjacent walls and windows to the outside, hence the outside weather can directly impact the room's environment.

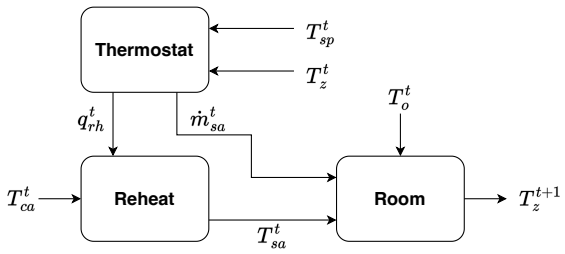

Fig. 3. Block diagram of the HVAC system used in the study.

A block diagram of the HVAC system for one room is depicted in Fig. 3. The thermostat block takes the current room temperature measurement T_z^t and the set-point T_{sp}^t to compute the control signals for the reheat (valve position q_{rh}^t) and the damper (resulting in the airflow rate $\dot{m}_{sa}^t$). The reheat block, under the controlled valve position q_{rh}^t and the conditioned air temperature T_{ca}^t from the air handling unit (AHU), produces the supply air temperature T_{sa}^t. The room block represents the room's air thermal dynamics which, given the supply air at rate $\dot{m}_{sa}^t$ and temperature T_{sa}^t, and under the disturbance of the ambient air temperature T_o^t, changes the room's air temperature to T_z^{k+1} at the next time step.

For each room, an experiment was conducted for 11 days to obtain data for model training and validation by uniformly and randomly changing the thermostat set-point of the room between $68\,^{\circ}\mathrm{F}$ to $74\,^{\circ}\mathrm{F}$ every 30 minutes. The set-points were changed programmatically using Python and a BACnet (building automation and control network) interface. The acquired dataset for each room was a set of time series data, which included the room temperature T_z, thermostat set-point temperature T_{sp}, supply air temperature T_{sa}, conditioned air temperature T_{ca}, reheat coil valve position q_{rh}, supply airflow rate $\dot{m}_{sa}$, and outside temperature T_o, measured at 5-minute intervals. Each dataset contained 3121 data points of the form $\{T_z^t, T_{sp}^t, T_{sa}^t, T_{ca}^t, q_{rh}^t, \dot{m}_{sa}^t, T_o^t\}_{t=1}^{3121}$. In the study, each dataset was split into a training set of N data points and a validation set of $3121 - N$ data points. We varied the size N of the training set and used small values for N to investigate the impact of the amount of training data on a model's performance and the data efficiency of each method.

B. Results for Room Models

Different ML methods, both physics-informed and non-physics-informed, were used to learn from the data a model of the room block for each room. To represent the thermal dynamics, an autoregressive model was used to predict the room temperature T_z^{t+1} at the next time step from a set of inputs, which include T_{sa}^t, $\dot{m}_{sa}^t$, and T_o^t, as $T_z^{t+1} = f_z(T_z^t, T_{sa}^t, \dot{m}_{sa}^t, T_o^t)$. For room 102, the input T_o^t is omitted because it does not affect the room's air temperature.

The following ML methods were evaluated:

- Non-physics-informed methods: Vanilla deep NNs were trained with different numbers of hidden layers (from 1 to 3) and different numbers of neurons (either 4, 8, 12, 16, or 32 neurons for each hidden layer), then the most accurate model on the validation set was selected. We also trained standard GPs with the SE kernel. In the result tables, these models are marked as `NN` and `GP` respectively.
- Physics-informed methods: From the physical laws of the room's thermal dynamics, the room model is increasing monotonic w.r.t. $\dot{m}_{sa}$ and T_{sa}. We therefore trained the following PIML models described in Sec. II: `Minimax NN`, `Lattice NN`, `approx-PIGP`, and `inducing-PIGP`. Furthermore, since the room's temperature should never change abruptly, we also trained a `Smooth NN` to promote smoothness of the model's output. The NN models were trained with different numbers of hidden layers (from 1 to 3) and different numbers of neurons (either 4, 8, 12, 16, or 32 neurons for each hidden layer). The GPs all used the SE kernel. The `approx-PIGP` models were trained with 8, 16, 32, 64, and 128 equidistant knots. The `inducing-PIGP` models were trained with 5, 10, 20, 40, and 80 inducing points. For each model type, the most accurate model on the validation set was selected.

For each room, these models were trained on different small training data sizes: $N \in \{16, 32, 64, 128\}$. For each N, the models were validated on the same validation set and two model performance metrics were calculated: the R^2 score and the root mean square error (RMSE). The results are reported in Table I and will be discussed in Sec. III-E.

C. Results for Reheat Models

Similarly, different ML methods were evaluated for predicting the supply air temperature produced by the reheat block from a set of inputs, which include q_{rh}^t and T_{ca}^t. Due to the reheat coil's dynamics, an autoregressive model was used: $T_{sa}^{t+1} = f_{rh}(T_{sa}^t, q_{rh}^t, T_{ca}^t)$. From the physical laws, this function is increasing monotonic w.r.t. T_{ca}^t and q_{rh}^t. Therefore, the same types of ML models as for the room block were trained and tested for different training data sizes $N \in \{32, 64, 128\}$, whose performance metrics are reported in Table II, which will be discussed in Sec. III-E. Because the reheat coil is not affected by the outside temperature disturbance T_o, the results are similar for both rooms and hence only the results for room 102 are reported.

D. Results for Thermostat Models

The thermostat model predicts the reheat valve position q_{rh}^t and the supply airflow rate $\dot{m}_{sa}^t$ from the room temperature T_z^t and the set-point temperature T_{sp}^t. Because the thermostat's control algorithm is unknown, the only applicable physical properties of it are that q_{rh} is bounded between 0% and 100% and $\dot{m}_{sa}$ is bounded between a minimum flow rate and a maximum flow rate, which vary from room to room but known in advance. Therefore, besides the non-physics-informed ML models `NN` and `GP`, we also trained the following PIML models described in Sec. II for the thermostat: `Soft-bound NN`, `Hard-bound NN`, `approx-PIGP`, and `inducing-PIGP`. The NNs were trained with different numbers of hidden layers (from 1 to 4) and different numbers of neurons (either 8, 16, 32, or 64

TABLE I

PERFORMANCE METRICS OF THE ROOM MODELS FOR ROOM 102 AND ROOM 106.

Room 102														
N	**NN**		**Smooth NN**		**Minimax NN**		**Lattice NN**		**GP**		**approx-PIGP**		**inducing-PIGP**	
	R^2	$RMSE$	R^2	$RMSE$	R^2	$RMSE$	R^2	$RMSE$	R^2	$RMSE$	R^2	$RMSE$	R^2	$RMSE$
16	-1.0114	4.8531	0.6762	1.0523	0.8562	0.7021	0.8819	0.6356	-3.3482	3.6193	0.8346	0.7521	0.8237	0.7751
32	-1.3318	5.3292	0.8829	0.6435	0.9642	0.3492	0.9978	0.0916	0.8975	0.5900	0.9972	0.0922	0.9974	0.0933
64	0.9983	0.0922	0.9983	0.0924	0.9982	0.0924	0.9982	0.0924	0.9984	0.0911	0.9982	0.0921	0.9981	0.0924
128	0.9897	0.1911	0.9904	0.1512	0.9982	0.0923	0.9985	0.0918	0.9982	0.0901	0.9983	0.0921	0.9981	0.0914
Room 106														
N	**NN**		**Smooth NN**		**Minimax NN**		**Lattice NN**		**GP**		**approx-PIGP**		**inducing-PIGP**	
	R^2	$RMSE$	R^2	$RMSE$	R^2	$RMSE$	R^2	$RMSE$	R^2	$RMSE$	R^2	$RMSE$	R^2	$RMSE$
32	-3.3482	7.2491	-1.1222	9.2167	-1.0312	6.1491	0.3642	1.4721	-2.8769	5.2164	-7.3889	5.3282	-8.9656	5.3322
64	0.7642	0.8980	0.7644	0.8982	0.9961	0.1022	0.8370	0.7642	-3.9984	3.2142	0.8511	0.7134	0.6551	1.0861
128	0.9241	0.5112	0.9243	0.5094	0.9967	0.0963	0.9804	0.2611	0.7798	0.8671	0.9835	0.2425	0.9912	0.0956

TABLE II

PERFORMANCE METRICS OF THE REHEAT MODELS FOR ROOM 102.

N	**NN**		**Smooth NN**		**Minimax NN**		**Lattice NN**		**GP**		**approx-PIGP**		**inducing-PIGP**	
	R^2	$RMSE$	R^2	$RMSE$	R^2	$RMSE$	R^2	$RMSE$	R^2	$RMSE$	R^2	$RMSE$	R^2	$RMSE$
32	0.9100	0.5551	0.9338	0.4754	0.9529	0.4034	0.9498	0.4351	0.9389	0.4742	0.8216	0.7823	0.7547	0.9162
64	0.9302	0.4913	0.9427	0.4432	0.9826	0.2442	0.9799	0.1642	0.9881	0.2012	0.9831	0.2431	0.9887	0.1943
128	0.9241	0.5094	0.9278	0.5131	0.9816	0.2512	0.9809	0.2522	0.9888	0.1942	0.9819	0.2451	0.9897	0.1931

TABLE III

PERFORMANCE METRICS OF THE q_{rh} OUTPUT OF THE THERMOSTAT MODELS FOR ROOM 102.

N	**NN**		**Soft-bound NN**		**Hard-bound NN**		**GP**		**approx-PIGP**		**inducing-PIGP**	
	R^2	$RMSE$	R^2	$RMSE$	R^2	$RMSE$	R^2	$RMSE$	R^2	$RMSE$	R^2	$RMSE$
32	0.8418	15.0153	0.8498	14.6674	0.8395	15.1281	0.9031	11.7532	-1.1241	55.4782	-2.2192	54.6400
64	0.9721	5.3962	0.9732	5.3644	0.9703	6.4572	0.9765	4.9512	0.9784	4.8245	0.9695	6.5462
128	0.9563	7.8663	0.9699	6.5022	0.9736	5.3214	0.9748	5.2145	0.9796	4.5193	0.9517	8.2741

TABLE IV

PERFORMANCE METRICS OF THE $\dot{m}_{sa}$ OUTPUT OF THE THERMOSTAT MODELS FOR ROOM 102.

N	**NN**		**Soft-bound NN**		**Hard-bound NN**		**GP**		**approx-PIGP**		**inducing-PIGP**	
	R^2	$RMSE$	R^2	$RMSE$	R^2	$RMSE$	R^2	$RMSE$	R^2	$RMSE$	R^2	$RMSE$
32	0.8543	14.9123	0.8572	14.7755	0.8395	15.5895	0.9087	11.4033	-1.3426	54.1934	-2.2183	54.0392
64	0.9828	4.3953	0.9839	4.1253	0.9695	6.5463	0.9863	4.0292	0.9852	4.2221	0.9713	6.2493
128	0.9815	4.9512	0.9820	4.8552	0.9517	8.2744	0.9881	3.9411	0.9818	4.8941	0.9611	7.4164

neurons for each hidden layer). The GPs all used the SE kernel. The `approx-PIGP` models were trained with 8, 16, 32, 64, and 128 equidistant knots. The `inducing-PIGP` models were trained with 5, 10, 20, 40, and 80 inducing points. For each model type, the most accurate model on the validation set was selected.

Different training data sizes were also considered for $N \in \{32, 64, 128\}$. The results for room 102 are reported in Table III (for q_{rh}) and Table IV (for $\dot{m}_{sa}$) and will be discussed in Sec. III-E. Because the thermostat is not directly affected by the outside temperature disturbance, the results are similar for both rooms and hence only the results for room 102 are reported.

E. Discussion

1) Benefits of physics-informed ML models: In general, integrating physical constraints into a ML model helps improve the model accuracy, especially for limited training data (i.e., small values of N). This is obvious from the results in the tables.

Take the room model results in Table I. For room 102, we can easily see that the integration of physical properties, particularly monotonicity in the `Minimax NN`, `Lattice NN`, `approx-PIGP`, and `inducing-PIGP` models, helped improve their accuracy compared with non-physics-informed models (`NN` and `GP`). For example, with just $N = 16$ training points, the performance metrics of `Smooth NN` ($R^2 = 0.6762$, $RMSE = 1.052$), `Minimax NN` ($R^2 = 0.8562$, $RMSE = 0.702$), `Lattice NN` ($R^2 = 0.8819$, $RMSE = 0.635$), `approx-PIGP` ($R^2 = 0.8346$, $RMSE = 0.7521$), and `inducing-PIGP` ($R^2 = 0.8237$, $RMSE = 0.7751$) are much better than those of the ordinary `NN` ($R^2 < 0$, $RMSE = 4.853$) and `GP` ($R^2 < 0$, $RMSE = 3.6193$). A similar observation can be made in the case of $N = 32$ training data points. However, the performance benefit of PIML diminishes for larger training data sizes. When there are more training data points ($N = 64$ and $N = 128$), integrating the physical properties into the models did not improve the accuracy remarkably.

For a more complex system, that is the room model of

room 106 which is directly affected by the outdoor weather, we found that integrating physical properties into the models boosted their accuracy significantly. Even with $N = 128$ training points, the performance metrics of `Minimax NN` ($R^2 = 0.9967$, $RMSE = 0.096$), `Lattice NN` ($R^2 = 0.9804$, $RMSE = 0.261$), `approx-PIGP` ($R^2 = 0.9835$, $RMSE = 0.2425$), and `inducing-PIGP` ($R^2 = 0.9912$, $RMSE = 0.0956$) are substantially better than ordinary `NN` ($R^2 = 0.9241$, $RMSE = 0.511$) and `GP` ($R^2 = 0.7798$, $RMSE = 0.867$). At a smaller training data size of $N = 64$ points, the `Minimax NN` model achieved R^2 score of 0.9961 on the validation dataset, which is significantly better than other models, especially non-physics-informed models. At even smaller training data sizes of $N = 16$ and $N = 32$, no models worked well due to the system's complexity and the influence of the outside weather disturbance.

Similar performance benefits of PIML methods over non-physics-informed methods can also be found in the reheat models (Table II) and the thermostat models (Tables III and IV). However, for the thermostat models, the accuracy improvements of PIML are less pronounced.

Our study clearly demonstrates that **PIML methods can improve both the data efficiency of learning and the model accuracy**, especially with limited training data and for complex systems, compared with non-physics-informed methods.

2) Comparison of the benefits of different physical properties: Different physical properties provide different benefits when incorporated into ML models. Our study examined various physical properties, in particular smoothness, boundedness, and monotonicity. Although the models with smoothness (`Smooth NN` in Tables I and II) and the models with boundedness (`Soft-bound NN`, `Hard-bound NN`, `approx-PIGP`, and `inducing-PIGP` in Tables III and IV) had better accuracy than non-physics-informed models, the differences were much less significant than the improvements achieved by the models with monotonicity. We can conclude that **not all physical properties are equally beneficial for PIML**, and integrating certain physical properties into ML can result in better model improvements than other properties. In our case, monotonicity is more beneficial than boundedness and smoothness.

3) Comparison of different model types: We observe that in many cases, the accuracy of physics-informed NN models was similar to or better than the accuracy of physics-informed GP models, especially when there was enough training data. One can conclude that, at least for our HVAC system, NN models are more favorable for PIML than GP models. However, given data from a system to be modeled, it is recommended that different PIML models and methods are evaluated on the same dataset to select the most suitable one.

4) Computational complexities of PIML methods: A direct quantitative comparison of the computational complexities of the ML methods studied in our work is not possible because of differences in their implementations. In particular, NN methods, both non-physics-informed and physics-informed, were implemented using PyTorch and were optimized for computational performance, whereas GP methods were implemented in standard Python and not optimized for performance. However, based on our experience implementing the methods and executing the experiments, we can make a qualitative comparison of their computational complexities.

Of the methods that we considered, in most cases, lattice NNs and minimax NNs provide better and more stable performance than the others. All NN-based methods are simpler to implement and much faster to train than GP-based methods. Physics-informed GP methods have the highest complexity and the longest training times. In addition, they are not as numerically stable as physics-informed NN methods. Therefore, for pragmatic reasons, in a real application, one should implement and evaluate NN-based PIML methods before more complex methods such as GP-based methods.

IV. Conclusion

We studied several physics-informed machine learning (PIML) methods for modeling HVAC systems using real experimental data and compared them in terms of model accuracy, data efficiency, integrated physical properties, and computational complexities. The considered methods include both NN-based methods and GP-based methods, in addition to ordinary non-physics-informed ML methods. Our study shows that, in general, integrating physical properties into ML models can improve significantly the accuracy, reliability, and data efficiency of ML, compared with non-physics-informed methods. The benefits, however, vary between different model types, physical properties, and system complexities. The results of this study will open more future work where physical constraints are relevant and model reliability and data efficiency are crucial factors. We plan to extend the current work to more PIML methods and physical properties, especially variants of neural networks such as recurrent neural networks and graph neural networks.

References

[1] D. Sturzenegger, D. Gyalistras, M. Morari, and R. S. Smith, "Model Predictive Climate Control of a Swiss Office Building: Implementation, Results, and Cost–Benefit Analysis," *IEEE Transactions on Control Systems Technology*, vol. 24, no. 1, pp. 1–12, Jan. 2016.

[2] E. T. Maddalena, Y. Lian, and C. N. Jones, "Data-driven methods for building control — A review and promising future directions," *Control Engineering Practice*, vol. 95, p. 104211, Feb. 2020.

[3] G. E. Karniadakis, I. G. Kevrekidis, L. Lu, P. Perdikaris, S. Wang, and L. Yang, "Physics-informed machine learning," *Nature Reviews Physics*, vol. 3, no. 6, pp. 422–440, May 2021.

[4] J. Drgoňa, A. R. Tuor, V. Chandan, and D. L. Vrabie, "Physics-constrained deep learning of multi-zone building thermal dynamics," *Energy and Buildings*, vol. 243, July 2021.

[5] E. Skomski, S. Vasisht, C. Wight, A. Tuor, J. Drgoňa, and D. Vrabie, "Constrained block nonlinear neural dynamical models," in *2021 American Control Conference (ACC)*. IEEE, 2021, pp. 3993–4000.

[6] H. Daniels and M. Velikova, "Monotone and partially monotone neural networks," *Trans. Neur. Netw.*, vol. 21, no. 6, p. 906–917, jun 2010.

[7] S. You, D. Ding, K. Canini, J. Pfeifer, and M. Gupta, "Deep lattice networks and partial monotonic functions," *Advances in neural information processing systems*, vol. 30, 2017.

[8] A. F. López-Lopera, F. Bachoc, N. Durrande, and O. Roustant, "Finite-dimensional gaussian approximation with linear inequality constraints," *SIAM/ASA Journal on Uncertainty Quantification*, vol. 6, no. 3, pp. 1224–1255, 2018.

[9] C. Agrell, "Gaussian processes with linear operator inequality constraints," *Journal of machine learning research*, vol. 20, no. 135, pp. 1–36, 2019.
[10] Z. Wang, T. Hong, and M. A. Piette, "Building thermal load prediction through shallow machine learning and deep learning," *Applied Energy*, vol. 263, 2020.
[11] A. Afram, F. Janabi-Sharifi, A. S. Fung, and K. Raahemifar, "Artificial neural network (ANN) based model predictive control (MPC) and optimization of HVAC systems: A state of the art review and case study of a residential HVAC system," *Energy and Buildings*, vol. 141, 2017.
[12] M. Behl, T. X. Nghiem, and R. Mangharam, "DR-Advisor: A Data Driven Demand Response Recommender System," in *Proceedings of International Conference CISBAT 2015 Future Buildings and Districts Sustainability from Nano to Urban Scale*, 2015.
[13] F. Bünning, B. Huber, P. Heer, A. Aboudonia, and J. Lygeros, "Experimental demonstration of data predictive control for energy optimization and thermal comfort in buildings," *Energy and Buildings*, vol. 211, 2020.
[14] T. X. Nghiem and C. N. Jones, "Data-driven demand response modeling and control of buildings with Gaussian Processes," in *2017 American Control Conference (ACC)*, 2017.
[15] A. Jain, T. X. Nghiem, M. Morari, and R. Mangharam, "Learning and Control using Gaussian Processes: Towards bridging machine learning and controls for physical systems," in *International Conference on Cyber-Physical Systems (ICCPS)*, 2018.
[16] G. Gokhale, B. Claessens, and C. Develder, "Physics informed neural networks for control oriented thermal modeling of buildings," *Applied Energy*, vol. 314, 2022.
[17] M. W. Ahmad, M. Mourshed, and Y. Rezgui, "Trees vs Neurons: Comparison between random forest and ANN for high-resolution prediction of building energy consumption," *Energy and Buildings*, vol. 147, 2017.
[18] C. Bishop, *Pattern Recognition and Machine Learning*, ser. Information Science and Statistics. New York: Springer-Verlag, 2006.
[19] C. E. Rasmussen and C. K. I. Williams, *Gaussian processes for machine learning*, ser. Adaptive Computation and Machine Learning series. London, England: MIT Press, Nov. 2005.
[20] J. Kotecha and P. Djuric, "Gibbs sampling approach for generation of truncated multivariate gaussian random variables," in *1999 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings. ICASSP99 (Cat. No.99CH36258)*, vol. 3, 1999, pp. 1757–1760.
[21] J. Riihimäki and A. Vehtari, "Gaussian processes with monotonicity information," in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, ser. Proceedings of Machine Learning Research, Y. W. Teh and M. Titterington, Eds., vol. 9. Chia Laguna Resort, Sardinia, Italy: PMLR, 2010, pp. 645–652.