Exploring LLM-Powered Agents for Modeling Thermal Dynamics of Buildings

Krzysztof Walczak
Amazon
Seattle, WA, USA
walkrzy@amazon.com

Mario Bergés*
Amazon
Seattle, WA, USA
Carnegie Mellon University
Pittsburgh, PA, USA
meberges@amazon.com

Abstract

Modeling building thermal dynamics is essential for energy optimization, yet building heterogeneity and non-stationary dynamics demand per-building customization that typically requires expert intervention. Automated scientific discovery workflows powered by Large Language Models (LLM) could significantly decrease the human expertise requirements for generating custom thermal models at scale, but their performance varies considerably depending on configuration and task, raising a key question: how should we configure scientific agents for thermal dynamics modeling? To answer this, we developed ThermalForge[1], a platform for experimenting with agentic approaches to hybrid neural-physics modeling of thermal dynamics. Using a year-long dataset from hundreds of U.S. residential smart thermostats, we investigate the impact of physics knowledge in prompts, constraints on neural architecture search, LLM effectiveness in determining physics-neural transition points, model stochasticity, cost-accuracy tradeoffs, and performance against benchmark methods.

Our results show that agentic modeling can match or exceed state-of-the-art automated methods while offering greater flexibility and reduced expertise requirements. We found important differences in configuring the physics-neural transition point, and demonstrate how the inherent variability of LLM outputs can be transformed from uncertainty into a mechanism for robust model discovery. This work presents the first large-scale application of LLM-powered modeling agents to real-world observational data, demonstrating their viability for thermal dynamics modeling and paving the way to scalable methods that embed domain expertise within automated workflows.

CCS Concepts

• **Computing methodologies** → **Modeling and simulation**; **Artificial intelligence**.

Keywords

Thermodynamic Models, Dynamical Systems, Modeling, Calibration, Digital Twins, Large Language Models, Machine Learning

ACM Reference Format:
Krzysztof Walczak and Mario Bergés. 2026. Exploring LLM-Powered Agents for Modeling Thermal Dynamics of Buildings. In *The 13th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation (BuildSys '26), June 22–25, 2026, Banff, AB, Canada.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3744256.3812580

1 Introduction

Modeling the thermal dynamics of buildings is essential for optimizing building energy use, yet widespread adoption of these models remains limited due to the human expertise requirements for their design and calibration. Physics-based approaches (e.g., [2, 9]) require experts to define and calibrate building-specific equations, and their accuracy is constrained by incomplete physics knowledge, computational resources, and measurable phenomena. On the other hand, data-driven/statistical approaches (e.g., [19, 46]) are easier to develop and more commonly employed by researchers in recent years, but they suffer from poor interpretability, unreliable out-of-distribution performance, and inefficient data use, while still requiring expertise for architecture design. Automated approaches have emerged recently to generate building-specific models (physics-based [24], or data-driven [35]) across large smart thermostat datasets by limiting model family choices and introducing heuristics in lieu of custom expert guidance for each model.

Recent developments in automated scientific discovery—where Large Language Models (LLMs) are being used to propose and evaluate hypotheses against experimental observations (e.g., [28], [18])—have opened up a new modeling paradigm that facilitates the incorporation of domain knowledge and meta-modeling decisions (e.g., selection of model family) while significantly reducing the human expertise required to automatically generate models of dynamical systems from data. This approach has produced promising results in other domains, though performance varies across modeling tasks and configurations.

Nonetheless, the pre-trained knowledge available through modern LLMs could potentially transform the way we scale up thermal modeling tasks, providing the expert guidance needed to truly customize models for each building given available data. To assess this, we designed ThermalForge, an agentic modeling framework for experimenting with this approach and better understand its benefits and drawbacks in the context of building thermal dynamics modeling.

*Mario Bergés holds concurrent appointments as an Amazon Scholar and as Professor of Civil and Environmental Engineering at Carnegie Mellon University. This paper describes work performed at Amazon and is not associated with Carnegie Mellon University.

[1]Code and Prompts: https://github.com/amazon-science/thermal-forge

BuildSys '26, Banff, AB, Canada

ACM ISBN 979-8-4007-2012-3/26/06
https://doi.org/10.1145/3744256.3812580

Our work is centered around answering the following question: *how should we design and configure scientific agents for the modeling and calibration of building thermal dynamics models?* We study this question in the context of hybrid neuro-physical models, using ThermalForge to experiment with different framework configurations. The framework is flexible enough to replicate design choices made by other LLM-powered agents such as HDTwinGen [18] and Scientific Generative Agents [28]. In particular, ThermalForge consists of a custom agentic framework that guides LLMs through parallel bi-level optimization processes in order to produce custom hybrid models for each building. For both physics-based and neural models, the outer-level optimization determines appropriate model structures (parametric equations or neural architectures) through LLM calls, and initializes them as differentiable PyTorch models, while the inner level calibrates the PyTorch model's parameters against observational data. The framework first optimizes the physics-based model until convergence (determined algorithmically or by LLM judgment), and then applies the same bi-level process to develop neural components for the residual patterns.

We explore the configuration space of ThermalForge using a publicly available year-long dataset from hundreds of U.S. residential smart thermostats. Specifically, we design experiments to study: (a) the impact of incorporating explicit physics knowledge in LLM prompts, (b) the effects of constraining neural architecture search to specific patterns, (c) the LLMs' effectiveness in determining optimal transition points between physics and neural modeling, (d) the stochasticity of the generated models, (e) cost-accuracy tradeoffs across framework configurations, and (f) comparative performance between custom per-home calibration versus a state-of-the-art automatic model calibration method. As a result, our paper makes the following contributions:

(1) We provide the first demonstration of applying LLM-powered modeling agents at scale, generating custom hybrid thermal dynamics models for 300 homes. To our knowledge this is the first large-scale empirical evaluation of agentic modeling solutions for complex dynamical systems in the wild.
(2) We rigorously characterize the configuration space for these modeling agents with respect to prompt engineering, token usage, performance, stochasticity and hyperparameters, resulting in recommendations for both practitioners and researchers alike.
(3) We demonstrate how to exploit the stochastic nature of modeling results to improve performance at lower costs.
(4) We release ThermalForge – the framework we used to explore LLM-powered modeling of dynamical systems to allow others to continue our exploration.

In summary, our work presents the first empirical evidence of the effectiveness of LLM-powered agentic approaches for thermal dynamics modeling, revealing sensitivities to architectural choices and trade-offs across settings. Overall, results indicate that ThermalForge can generate sophisticated thermal dynamics models that match the accuracy of other automated workflows in the literature while remaining flexible to the choice of model family and available sources of data/knowledge for each building. Our experiments with different design choices reveal practical insights for applying these frameworks at scale, and thereby unlock the latent thermal dynamics modeling expertise available in modern LLMs.

The rest of the paper is organized as follows: we first provide an overview of thermal dynamics models and automated modeling workflows in Section 2. Then, in Section 3 we introduce our thermal dynamics modeling approach in more detail, formalize it into an algorithmic framework and describe the design choices that we consider worthy of exploration during our experiments. Section 4 describes the experimental plan and results, as well as the dataset that made them possible. Lastly, Section 5 discusses the implications of our experimental results and provide recommendations for future work.

2 Background and Prior Work

We will consider a thermal dynamics model to be a mathematical function h that computes spatially-averaged dry-bulb indoor air temperature values at a future time point given information about current/past temperature values, outdoor weather and internal loads. In its discrete-time representation, these models have the following general form:

$$\mathbf{x}_{n+1} = h_{\boldsymbol{\theta}}(\mathbf{x}_n, \mathbf{u}_n, \mathbf{w}_n) + \boldsymbol{\epsilon}_n \tag{1}$$

Here, h_θ is the dynamics function of the system (parametrized by $\boldsymbol{\theta}$), $\mathbf{x_n}$ is the state vector at time-index n, $\mathbf{u_n}$ represents the vector of controllable inputs/forcings acting on the building (e.g., cooling/heating systems, fans, etc.), and $\mathbf{w_n}$ represent the uncontrollable disturbances/forcings (e.g., outdoor temperature, solar gains, occupancy). Notably, one crucial assumption this model makes is that the system is time-invariant – something that is rarely true in real buildings. We will also note that h can be any family of functions, although for clarity we will use $f_{\boldsymbol{\theta_p}}$ for physics-based models and $g_{\boldsymbol{\theta_n}}$ to refer to neural-network or black-box models of the dynamics. Similarly, h can be any arbitrary composition function C to combine physics-based and neural models, i.e. $h = C(f_{\boldsymbol{\theta_p}}, g_{\boldsymbol{\theta_n}})$. Lastly, modeling choices and available data result in different $\mathbf{x}$, $\mathbf{u}$ and $\mathbf{w}$ definitions. In our experiments, for example, the state $\mathbf{x}$ is a vector of indoor temperatures at different zones/rooms ($\mathbf{T_i}$), our controllable inputs $\mathbf{u}$ are heating duty-cycles ($\mathbf{Q}$), and the disturbances $\mathbf{w}$ are outdoor temperature values ($\mathbf{T_o}$).

To better situate our problem with respect to the relevant body of work, we now review the literature on the specific models $h_{\boldsymbol{\theta}}$ that have been proposed, the tools and workflows that have been designed to automate the design and calibration of these functions, the knowledge that LLMs have shown to have to assist in this automation process, and the field of automated scientific discovery.

2.1 Models of Building Thermal Dynamics

A variety of models have been proposed for Equation 1 and they can be generally categorized as being either white, gray or black box by virtue of their interpretability and transparency [4]. In this section, we review each of them in more detail. The goal of this review is not to be comprehensive but rather to shed light on the state-of-the-art modeling approaches and their comparative reliance on human expertise. For a more thorough and up-to-date review of the modeling approaches and tools, we refer the reader to other sources such as [10, 19, 25].

White Box. Fully physics-based models include those that are contained in building energy simulation tools such as EnergyPlus [32] and TRNSYS [38]. In these models, the parameters $\boldsymbol{\theta}$ as well as the structure of the function f itself are typically established by an expert modeler and some manual/indirect calibration when observations are available. These models are usually based on coupled partial differential equations for the different physical phenomena being simulated, which are expensive to formulate, discretize and/or solve but can be very accurate, interpretable and generalize well to samples outside of the calibration/training distribution.

Grey Box. As their name implies, these models typically rely on reduced-order and/or lumped representations of the system and its dynamics, which can be more efficiently calibrated using observational data. One of the most commonly used models in this category are Resistance Capacitance (RC) thermal network models [23]. These models substitute the spatially-varying building elements with lumped thermal capacitance and resistance nodes, effectively turning partial differential equations into ordinary ones. The modeler chooses the configuration of the thermal components and their connections (i.e., $f_{\boldsymbol{\theta}}$) and then identifies the model parameters $\hat{\boldsymbol{\theta}}$ with data. These models require less fine-tuning and expert assistance than white box ones, but their performance varies across building components [21], they can underestimate energy use [14] and generally do not benefit from additional calibration data past a certain point [1, 3].

Black Box. The last category, also referred to as data-driven or statistical/neural models, are models whose structure and parameters need not bear any connection to the physical process we are modeling. Instead, their specification is driven entirely by their ability to accurately predict the dynamics observed in the available data. Modelers define the specific structure of the function $g_{\boldsymbol{\theta_n}}$ (e.g., the architecture of a neural network), and calibrate its parameters with available data. Statistical auto-regressive models (e.g., ARX [16]) and various neural network architectures (e.g., LSTM [13, 19]) are popular choices within this class of models. Though they generally outperform grey-box and white-box models [3, 19], their ability to generalize outside the training data distribution or to offer valuable insights to building operators is limited. To that end, some researchers have focused on embedding physics-constraints and knowledge into black-box models such as by constraining the architecture and function blocks for deep neural networks and/or their outputs (e.g., [10, 45]).

Hybrid Models. Another class of approaches combine white-box and black box models in complimentary ways (e.g., [29]). However, in addition to the expertise requirements brought about by each of the modeling paradigms being combined, these hybrid approaches also need expertise to define the composition function C. In particular, we note that there is a class of hybrid models (sometimes referred to as Latent Force Models) that explicitly combine grey-box approaches for $f_{\boldsymbol{\theta}}$ and black-box approaches to approximate the time-varying residual $\boldsymbol{\epsilon_n}$ that seem particularly promising [17] though they require the modeler to choose, *a priori*, the specific model family for the grey-box and black-box approaches.

2.2 Automated Modeling Workflows

Even with access to specialized software toolboxes for specifying and calibrating any of the aforementioned models of building thermal dynamics (e.g., MPCPy [5], BLDG [22]), scaling these efforts to hundreds or thousands of buildings remains a challenge due to their reliance on human expertise. To alleviate these concerns, recent research has focused on developing tools for automating the full model generation/calibration process. In [41] and [24], for example, researchers introduce automated modeling approaches where the complexity of a gray-box model is optimized via a forward- (and/or backward-) selection process that stops based on some statistical criteria (e.g, Bayesian Information Criteria or likelihood ratio tests). Notably, the automated method from [24] has been successfully applied to generate accurate models out of a dataset of 60,000 smart thermostats [43]. While all of these research efforts share great similarity with our overall goal of scaling-up the modeling efforts at lower costs, they are (by design) committed to a specific modeling approach (e.g., lumped RC models) and would require non-trivial efforts to allow the models to benefit from additional data sources, knowledge or to modify the modeling approach altogether. Our hypothesis is that all of these higher-level modeling decisions typically reserved to the modeling expert could –in principle– be made by LLM agents.

In a separate line of research, transfer learning approaches have been receiving significant attention as of late [7, 20, 26, 34] as another way to alleviate expertise requirements for modeling at scale. In [35], for example, researchers introduced an LSTM model pre-trained on 450 homes paired with a transfer learning approach to fine-tune it to new homes using small amounts of data thereby simplifying the modeling process. While promising, these approaches have only been tested in the context of simulated data, and additional expert guidance may be needed to improve performance on more diverse and realistic data distributions.

In summary, despite efforts to reduce or eliminate human expertise from the modeling process, many decisions still require expert input especially if models need to be specialized for each building to leverage different modeling paradigms and the available knowledge and data sources.

2.3 Foundation Models for Scientific Discovery

Using empirical observations/data to learn the dynamics function h_θ and/or the constitutive equations that are part of it are tasks associated with the scientific discovery process. Though traditional machine learning and statistical methods dominated the research on automating these tasks [6], in recent years there has been a flurry of research projects attempting to leverage LLMs for these efforts. In contrast with approaches that automate a particular modeling workflow (like those described in the previous section) these LLM-powered processes are significantly more adaptable owing to the broad pre-trained knowledge, flexible language interface and reasoning capabilities of modern LLMs. For example, LLMs have been incorporated into automated frameworks to discover/learn constitutive/dynamics models in materials science [40], physics [15], biomedical systems [30], and ecology [18]. And more recently, other kinds of foundation models beyond LLMs have been proposed to infer governing dynamics from time-series data (e.g., [31, 44]).

Overall, there is a growing body of work on scientific "agents" based on LLM-powered workflows and so-called agentic systems. Within this area, two prior efforts serve as the basis for our design of ThermalForge: Scientific Generative Agent [28] and HDTwin-Gen [18]. These approaches propose a bi-level optimization framework that leverages LLMs to propose both candidate models and their improvements after empirically calibrating them with data. These frameworks were tested on simulated datasets of cancer growth, epidemics, ecological populations, molecule design and material science; as well as a real-world dataset of predator/prey population dynamics. Though they produced promising results, we note that they have not yet been tested at scale on real datasets with noisy observations of non-stationary dynamics such as those exhibited by smart thermostats. Moreover, the power of these scientific agents comes at the cost of an almost-infinite parametrization (e.g., the space of possible LLM prompts) and potential brittleness due to the stochastic nature of LLM outputs. Our experiments are designed to assess this high-dimensional design space of configurations for scientific agents and provide practical guidance on how they could be best applied to scale up thermal dynamic modeling efforts.

3 Approach

A human expert conducting modeling of building thermal dynamics goes through an iterative two-step process: 1) model (hypothesis) proposition, and 2) model parameter calibration against observational data. Thus, a common approach to imitating these steps in a modeling agent is to frame them as a bi-level optimization problem [18, 28]. In the outer optimization the goal is to find a function f, which is parametrized by θ. Using f, in the inner optimization we estimate the best-fitting value of its parameters ($\hat{\theta}$). The search for f and θ is guided by a loss function $\mathcal{L}$. Equation 2 summarizes this:

$$\min_{f,\Theta} \mathcal{L}\big(f, \Theta, \hat{\theta}\big) \tag{2a}$$

$$\text{s.t.} \quad G\big(f, \Theta\big) \leq 0 \tag{2b}$$

$$\hat{\theta} \in \underset{\theta \in \Theta}{\operatorname{argmin}}\, \mathcal{L}\big(\theta; f\big) \tag{2c}$$

Here $G(\cdot) \leq 0$ represents the correctness of the generated solution and Θ represents the set of all possible values for θ.

Each bi-level optimization routine consists of a loop in which, at each iteration k, an LLM is prompted to generate a new function f and its parameter set Θ based on its internal knowledge and results from prior generations. In other words, the outer loop (Eq. 2a) is solved via:

$$f, \Theta = LLM\big(\{f^k, \mathcal{L}^k, \theta^k\}_{k \in K}; P\big), \tag{3}$$

where K is the cardinality of the set of solutions from the previous generations, and prompt P instructs the LLM to conduct thermal dynamics modeling for a residential building. Initially, $K = 0$ and the LLM generates the first hypothesis with only the instructions in P. To ensure that our model is differentiable, the LLM is requested to generate a PyTorch implementation of function f that complies with $G(\cdot)$ (e.g., the PyTorch module can be optimized). The inner optimization (Eq. 2c) searches for optimal parameters $\hat{\theta}$ via a PyTorch optimizer, given the differentiable PyTorch model from the outer optimization. The next section describes this implementation in more detail.

3.1 Framework

Since our primary research question focuses on how we should use the scientific agents in thermal dynamics modeling and calibration, we study Equation 2 under the more complex modeling scenario of hybrid neuro-physical models. In this setting we can explore LLMs' capabilities to exploit their internal knowledge to propose customized physics-based thermal models, perform neural architecture search, and, most importantly, integrate these two modeling approaches into a holistic solution. Whereas the HDTwin-Gen framework [18] is explicitly targeting these types of hybrid models, we design ThermalForge to allow us to more finely control the way agents make decisions about the composition function C used to combine physics and neural models. Figure 1 illustrates the resulting ThermalForge framework. At a high-level, it consists of two phases: physics-based and neural modeling. In each of them, we apply the previously introduced bi-level optimization approach to generate the corresponding dynamics models.

Physics-Based Modeling Phase: Each iteration of the physics-based design phase begins with an LLM call to generate a thermal dynamics model proposal. We use prompt P_{pg} and feedback from previous generations Z_p to instruct the LLM to generate new model proposals. This results in PyTorch code that implements function $f^k_{\theta_p}$, where k is the iteration number. Additionally, we obtain the set of parameters Θ^k_p for the inner optimization. Then, during the inner optimization, we calibrate the model with observational data $\mathcal{X}_j$ (for house j). This data includes indoor temperature T_i (from one or multiple sensors), outdoor temperature T_o, and heat input Q. The model parameters are optimized to minimize a loss function $\mathcal{L}$. We use mean squared error (MSE) between predicted and observed indoor temperatures as our loss function. The inner optimization results in parameter estimates $\hat{\theta}^k_p$ and corresponding loss value $\mathcal{L}^k_p$.

The convergence check and subsequent actions are determined by parameter δ, which controls the agent's decision-making capability. An important distinction between a workflow and an agent is that a workflow follows a predefined sequence of steps, while an agent autonomously decides its actions based on feedback and intermediate results. If the agentic flow is enabled ($\delta = 1$), we use LLM to decide on the next step based on the results from the previous k iterations: $f^{1:k}_{\theta_p}$, $\hat{\theta}^{1:k}_p$, and $\mathcal{L}^{1:k}p$. The LLM instructions for this check are provided in prompt P_{pa}. If the agentic flow is disabled ($\delta = 0$), we check if we reached the maximum number of generations, M_p.

If the convergence check indicates continuation of physics-based model design, we use LLM to generate feedback (*Feedback Elicitation*) for the next generation. This feedback, intended to improve future results, summarizes the results generated so far. Using prompt P_{pz}, the LLM assesses what worked well in the previous generations and what did not. Unlike other studies that apply evolutionary selection (e.g., [28]), we maintain all solutions generated by the algorithm, as our experiments showed no improvement from various evolutionary selection mechanisms. If the convergence check

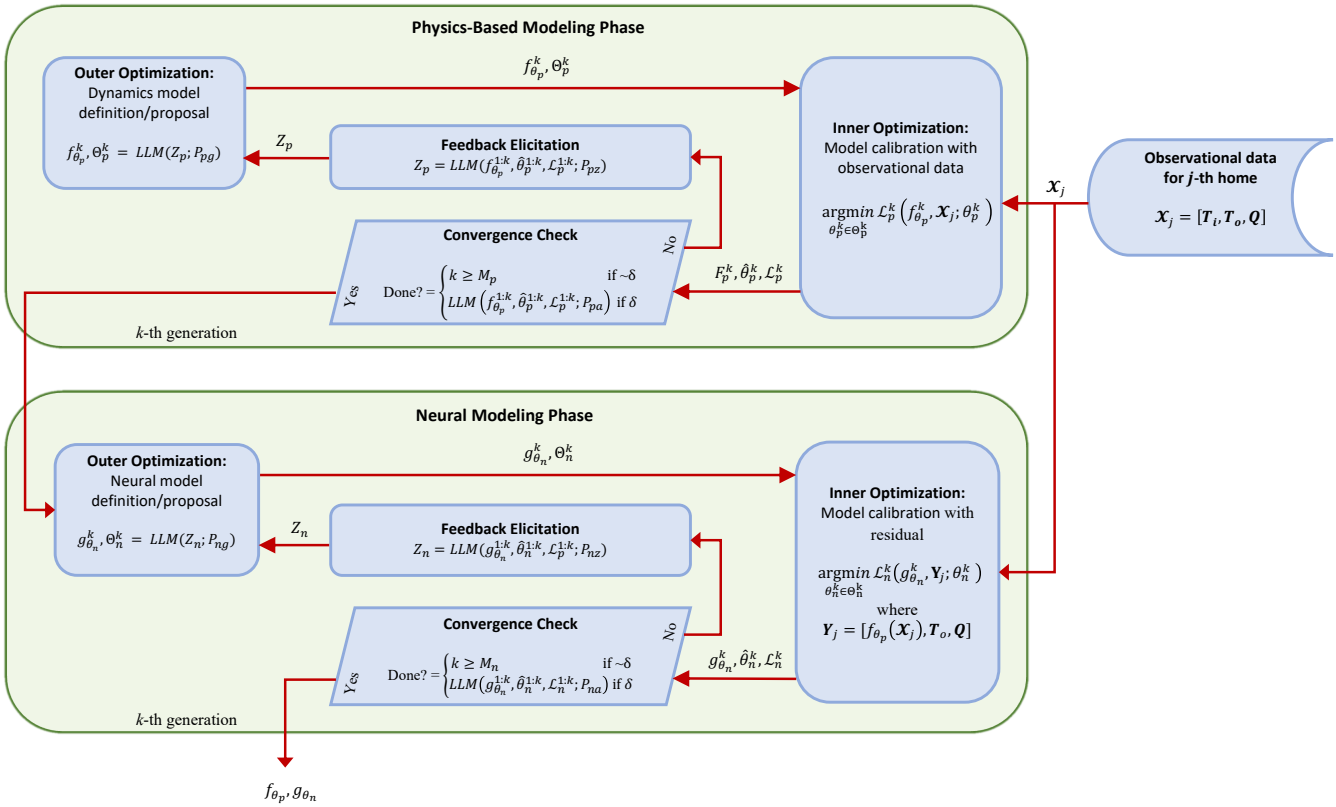

Figure 1: The ThermalForge - Thermal Dynamics Modeling Framework

indicates completion of the physics-based modeling phase, ThermalForge selects the best model f_{θ_p} from all k generations based on the validation loss. Specifically, it chooses the model that achieved the lowest validation loss.

Neural Modeling Phase: The procedure for the next phase (neural modeling) follows the same general steps as the physics-based modeling, though with several important nuances. First, all prompts are adapted to provide LLM with instructions specific to generating neural models for thermal dynamics. Second, we feed the neural model with the output from the physics-based model $f_{\theta_p}(\mathcal{X}_j)$, enabling it to model only the residual between physics-based predictions and target observational data. A key distinction lies in the prediction horizon: while the physics-based model makes predictions one time step at a time (e.g., 1 hour), the neural model receives physics-based model outputs for the prediction horizon (e.g., 1 day roll-outs of f_{θ_p}). This approach improves residual modeling accuracy and enables the use of convolutional architectures, which are well suited for time series modeling. Finally, the neural modeling phase outputs PyTorch models implementing two functions: the physics-based model f_{θ_p} and the neural model g_{θ_n}.

We present the algorithmic description of the thermal dynamics modeling agent in Algorithm 1. The complete set of prompts is provided in the GitHub repository. An example temperature prediction produced by a model generated with ThermalForge is shown in Figure 2, where we present the interquartile range (IQR) and

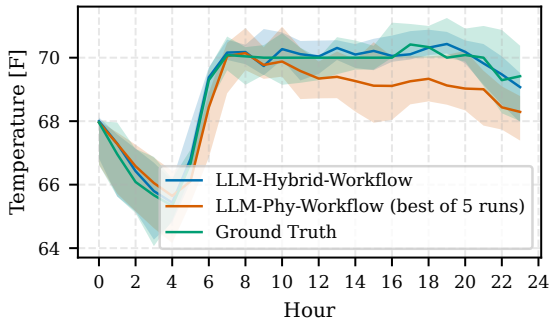

Figure 2: Example of median hourly indoor temperature prediction for a single house over 24 hours. Shaded areas cover the inter-quartile range (IQR).

median values for 1-day indoor temperature predictions comparing physics-based and hybrid models in a single home.

Importantly, we note that though our work focuses on thermal dynamics modeling, nothing in the framework is specific to this type of systems. Thus, future extensions of our work can focus on testing and refining the framework for modeling other dynamical systems.

Algorithm 1 Modeling Agent for Thermal Dynamics

Input: T_i, T_o, Q: observational data; δ: enable agent to decide on its state transitions; M_p, M_n: max number of generations; P_{pg}, P_{pz}, P_{ng}, P_{nz}, P_{pa}, P_{na}: prompts
Output: f_{θ_p}: physics-based model, g_{θ_n}: neural model

// Physics-based modeling
$modelGen \leftarrow$ **true**
$Z_p \leftarrow None$
$k \leftarrow 0$
while $modelGen$ **do**
  $k \leftarrow k + 1$
  // Generate model
  $f^k_{\theta_p}, \Theta^k_p \leftarrow LLM(Z_p; P_{pg})$
  // Optimize model
  $\hat{\theta}^k_p, \mathcal{L}^k_p \leftarrow optimize(f^k_{\theta_p}, \theta^k_p, T_i, T_o, Q)$
  // Generate feedback
  $Z_p \leftarrow LLM(f^{1:k}_{\theta_p}, \hat{\theta}^{1:k}_p, \mathcal{L}^{1:k}_p; P_{pz})$
  if δ **then**
    // Agent decides on state transition
    $modelGen \leftarrow LLM(f^{1:k}_{\theta_p}, \hat{\theta}^{1:k}_p, \mathcal{L}^{1:k}_p; P_{pa})$
  else
    if $k \geq M_P$ **then**
      $modelGen \leftarrow$ **false**
    end if
  end if
end while
$f_{\theta_p} \leftarrow selectBest(f^{1:k}_{\theta_p}, \mathcal{L}^{1:k}_p)$
// Neural modeling
$modelGen \leftarrow$ **true**
$Z_n \leftarrow None$
$k \leftarrow 0$
while $modelGen$ **do**
  $k \leftarrow k + 1$
  // Generate model
  $g^k_{\theta_n}, \Theta^k_n \leftarrow LLM(Z_n; P_{ng})$
  // Optimize model
  $\hat{\theta}^k_n, \mathcal{L}^k_n \leftarrow optimize(g^k_{\theta_n}, \theta^k_n, f_{\theta_p}, T_i, T_o, Q)$
  // Generate feedback
  $z \leftarrow LLM(g^{1:k}_{\theta_n}, \hat{\theta}^{1:k}_n, \mathcal{L}^{1:k}_n; P_{nz})$
  if δ **then**
    // Agent decides on state transition
    $modelGen \leftarrow LLM(g^{1:k}_{\theta_n}, \hat{\theta}^{1:k}_n, \mathcal{L}^{1:k}_n; P_{na})$
  else
    if $k \geq M_N$ **then**
      $modelGen \leftarrow$ **false**
    end if
  end if
end while
$g_{\theta_n} \leftarrow selectBest(g^{1:k}_{\theta_n}, \mathcal{L}^{1:k}_n)$

3.2 Design Choices

Thermal dynamics model generation with LLMs involves several critical design choices that affect its performance. First, the extent of physics knowledge encoded in prompts (P_{pg} in Figure 1) may influence the quality of generated models as shown in prior work (see e.g., Appendix H.7 in [18]). This raises questions about the optimal balance between providing domain expertise and allowing the LLM to explore solution spaces independently. Notably, the sensitivity to prompt design seems to be strongly task-dependent, suggesting that prior results may not translate to the thermal dynamcis modeling task.

The neural architecture design presents another crucial consideration (prompt P_{ng} in Figure 1) that has not been previously studied. By tuning P_{ng} in ThermalForge, we can either constrain the search space to proven architectures with hyperparameter optimization, allow modification of predefined architecture through code generation, or enable completely unconstrained neural architecture search. Each of these approaches offer different trade-offs between exploration flexibility and solution reliability.

The transition strategy between physics-based and neural modeling phases, controlled by the δ parameter in Figure 1, represents yet another important design choice. A workflow-based approach with predefined iteration counts offers predictability but may not adapt to specific modeling challenges. In contrast, an agent-based approach using LLM for decision-making could potentially optimize this transition based on intermediate results. Crucially, exposing this parameter to the user in ThermalForge allows us to study the transition in much more detail than previous studies.

In addition to these, another key practical consideration is the computational cost of generating models with this process. We analyze LLM token consumption during model generation to add another dimension to our final assessment of the different configurations of ThermalForge (in addition to the predictive performance of the resulting models).

Finally, we consider the stochasticity of the generated models, stemming from the nature of the process used to produce LLM outputs (controlled by the temperature) as well as randomness introduced through the optimizers used for the inner level optimization routine. We analyze how this variability affects the performance of the generated thermal dynamics models.

In the following section, these design choices will be systematically evaluated in our experimental analysis to provide insights into their impact on model performance and practical applicability.

4 Experiments

Based on the design choices we previously described, we create experiments to better characterize the design space for the framework. We evaluate it using a year-long data from residential smart thermostats that is publicly available [27]. This dataset is a subset of the larger Donate-Your-Data initiative from smart thermostat manufacturer Ecobee [12] and includes indoor and outdoor temperatures, HVAC run time (stages 1 to 3), and fan run time for each home at a 5-minute resolution for a full year. From an initial dataset of 1,000 homes, we selected 300 that contained at least 180 heating days to ensure sufficient data for robust model training and validation. We conduct experiments only on those days. In 230

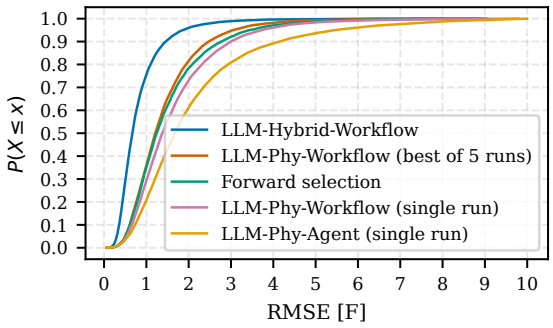

Figure 3: CDF of RMSE for different design choices

homes, there is more than one temperature sensor, and we use all available sensors in our experiments. For each home, we split the dataset on a daily basis into training (80%), validation (10%), and test (10%) sets. Our models predict temperature at 1-hour intervals for a 1-day horizon. As our primary evaluation metric for the accuracy of our models, we use the root mean squared error (RMSE) between the predicted 24-hour vector and the ground truth from the dataset. Next, we compute the cumulative distribution function (CDF) across all homes to assess the design choices we explore in our experiments. Figure 3 shows the plots that characterize our experimental results combining RMSE and CDF values. Specifically, for a given RMSE value, the corresponding vertical-axis value in the line corresponds to the proportion of 24-hour predictions across all data in the test set that resulted in equal or lower RMSE (in Fahrenheit). Hence, the sooner the curve saturates, the better the model. We will refer to specific values on this curve by indicating the percentile of RMSE we are discussing (e.g., P90 for the 90th percentile). It is also worth pointing out that, as a benchmark, other models trained on similar datasets in the literature report RMSE values of approximately 2.5 Fahrenheit for 24-hour predictions [35, 42] and 1.2 Fahrehneit for 6-hour predictions [19], though it is difficult to make precise comparisons due to the many different modeling, dataset and configuration choices involved.

Our implementation uses Anthropic's Claude Sonnet 4.5 LLM. We evaluated different LLM temperature parameter settings (0.2, 0.5, 0.8), but did not observe a major impact on the performance of generated models. We run each home's experiment on a separate machine. This parallel execution across 300 instances enables completion of a single experiment in approximately one hour.

4.1 Benchmark: Forward Model Selection

To evaluate the performance of LLM-based model generation, we compare it against the automated forward model selection method introduced in [24]. This method provides a systematic, data-driven approach for identifying optimal thermal models for residential buildings. The forward model selection procedure begins with the simplest first-order RC model and iteratively increases complexity by adding model components. At each iteration, the method evaluates potential extensions from a predefined set of thermal components and selects the component that provides the most statistically significant improvement based on likelihood ratio tests. These thermal components represent distinct physical aspects of building dynamics:

- T_i : effective indoor air temperature
- T_m : effective temperature of the interior thermal mass
- T_e : effective temperature of the building envelope mass
- T_h : effective temperature of the heater(s)
- T_s : effective temperature of the sensor(s)

The procedure terminates when no additional component yields a significant improvement, preventing overfitting while ensuring adequate model complexity. This automated approach has been successfully validated on 247 Dutch residential buildings and subsequently on 60,000 North American homes using smart thermostat data [43], demonstrating its capability to scale across heterogeneous building stocks and geographical regions without requiring manual expert intervention.

We use this forward model selection method as our baseline to assess whether LLMs can generate comparable or superior thermal dynamics models for residential buildings while potentially offering advantages in terms of flexibility, extension to hybrid models, or computational efficiency. We apply forward model selection to the dataset used in our experiments to ensure the same evaluation environment. The CDF of RMSE obtained with this method is shown as "Forward selection" in Figure 3.

4.2 Physics Modeling Knowledge in Prompts

In this experiment, we explore how physics knowledge in the prompt P_{pg} affects thermal dynamics modeling results. We extend the basic prompt with expert-recommended modeling strategies. To incorporate domain-specific thermal modeling expertise into our LLM-powered modeling framework, we developed a structured prompt based on the grey-box modeling methodology introduced in [24]. The prompt specifies a stochastic continuous state-space formulation where building thermal behavior is represented through up to five temperature states: interior air temperature (T_i), sensor temperature (T_s), interior thermal mass temperature (T_m), building envelope temperature (T_e), and heater temperature (T_h). Each state is associated with an equivalent thermal capacitance C, while thermal resistance R defines heat transfer between nodes. Following the approach in [24], the prompt provides the differential equations governing heat transfer dynamics, including stochastic terms ($\sigma d\omega$) representing unmodeled phenomena. The prompt explicitly instructs the LLM to construct models using combinations of these physics-based states rather than black-box approaches like neural networks. Additionally, the prompt adapts the modeling framework to accommodate multi-sensor configurations by instructing the LLM to exploit inter-zone heat transfer when multiple temperature sensors are available. This physics-informed structure guides the LLM to generate models that respect fundamental thermodynamic principles while allowing flexibility in model complexity — from simple first-order models to higher-order representations that better capture real building thermal dynamics. The full prompt is provided in the GitHub repository.

Results: Though not shown in Figure 3, our results indicate that providing additional information in the prompt did not result in significant differences in model performance (P90 of RMSE is 3.04 F for basic prompt, and 3.23 F for expert prompt). The CDF of RMSE

obtained with expert knowledge in the prompt is nearly identical to that without expert knowledge, shown as "LLM-Phy-Workflow (single run)" in Figure 3. Through analysis of the generated code, we found that the LLM follows the expert guidelines from the prompt in the generated models. However, using more basic prompts the LLM finds alternative models that are equally effective.

4.3 Neural Architecture Search

In the previous section, we examined how expert guidance impacts physics-based modeling results. We now turn our attention to neural modeling of the residuals, for our hybrid approach. We explore three options: 1) hyperparameter optimization for a predefined architecture, 2) code generation for a predefined architecture, and 3) unconstrained neural architecture search. We use the U-Net network [36] as our predefined architecture, which has demonstrated strong performance in various time series analysis tasks (e.g., [33, 39]). The objective behind these design choices parallels our physics-based modeling investigation, i.e., to explore how additional expert knowledge impacts the results. In option 1, we do not generate neural model code. Instead, the LLM is tasked with optimizing two hyperparameters: the number of encoding blocks and the number of output channels in the first layer of the U-Net architecture. In option 2, we remove the constraint of predefined code and use the LLM to generate U-Net model code. Finally, in option 3, we instruct the LLM to perform full architecture search without any constraints or suggestions for a specific architecture. It is expected to find an architecture that best fits our hybrid approach to thermal dynamics modeling.

Results: We found negligible differences among these three neural architecture search options (P90 of RMSE is 1.47 F for full architecture search, 1.51 F for U-Net code generation, and 1.75 F for U-Net hyperparameter optimization). The CDF of RMSE for the full architecture search option is shown in Figure 3 as "LLM-Hybrid-Workflow". As this is our best performing model, we further analyzed the RMSE distribution for individual homes. Figure 4 shows the IQR and median values of RMSE for both hybrid and physics-based models. The homes are sorted by median RMSE for the physics-based model. We observe that hybrid modeling not only reduces overall RMSE but also decreases the spread of RMSE values, resulting in more consistent performance across individual homes.

4.4 Transition from Physics to Neural Modeling

In this section, we compare the modeling results of setting the agentic flow, controlled by the δ parameter in Algorithm 1.

For the workflow implementation ($\delta = 0$), we predefine the number of iterations: 10 for physics-based modeling and 5 for neural modeling. For the agent implementation ($\delta = 1$), we use the LLM to determine when to transition from physics-based to neural modeling, and when to complete the neural model design. We instruct the LLM to make these decisions based on previously generated models and their corresponding validation losses. Specifically, we prompt the LLM to conclude a particular step if no improvement is observed over several generations (e.g., 5). The full prompts (P_{pa} and P_{na}) are provided in the GitHub repository.

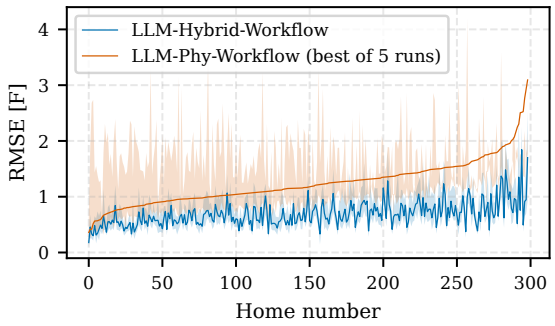

Figure 4: Median of per-home RMSE. Shaded areas cover the inter-quartile range (IQR).

Results: Figure 3 shows the results for both the workflow and agent approaches for the physics-based model, shown as "LLM-Phy-Workflow (single run)" and "LLM-Phy-Agent (single run)". We observe that the workflow outperforms the agent. Analysis reveals that the agent transitions from physics-based modeling prematurely compared to the workflow's predefined 10 generations. The results also indicate that additional iterations and LLM-generated feedback improve model performance. We recognize that the agent's behavior could potentially be improved with more refined prompting, and we leave this investigation for future research. For the hybrid model, both approaches achieve similar performance (P90 of RMSE is 1.4718 F for workflow and 1.4707 for agent).

4.5 Stochasticity Analysis

A notable characteristic of LLM-powered model generation is the inherent stochasticity in the output, which manifests as significant variability in model performance across multiple runs for the same building. Figure 5 illustrates this phenomenon by showing the P90 of RMSE across 5 physics-based model generations for all 300 homes in our evaluation dataset. The spread between minimum and maximum P90 values reveals substantial run-to-run variation, with some homes exhibiting differences of several degrees Fahrenheit in their prediction errors.

This stochasticity is partially attributable to the non-deterministic nature of LLM inference, where temperature sampling and the probabilistic selection of tokens lead to different model architectures, parameter initializations, and physical assumptions across runs, even when provided with identical prompts and building data. While this variability might initially appear as a limitation, we demonstrate that it can be systematically exploited to improve model reliability and performance.

Our approach leverages a validation set to select the best performing model from multiple generated candidates for each home. This selection strategy yields substantial improvements in test set performance compared to relying on a single model generation. Figure 3 shows the improvement achieved through this mechanism, comparing "LLM-Phy-Workflow (best of 5 runs)" vs. "LLM-Phy-Workflow (single run)". Critically, we observe that models underperforming in a single run converge to comparable accuracy levels when this

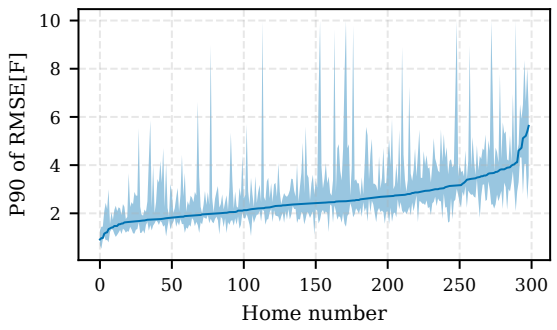

Figure 5: P90 of RMSE across 5 physics-based model generations. The line represents the median P90 value, and the shaded area spans the minimum to maximum P90 values.

multi-run selection approach is applied. For instance, agentic control represented by "LLM-Phy-Agent (single run)" in Figure 3 underperforms when compared to "LLM-Phy-Workflow (single run)". However, selecting the best model from multiple runs with the agentic control leads to nearly identical results as those shown in "LLM-Phy-Workflow (best of 5 runs)". This convergence suggests that the stochasticity is not indicative of fundamental modeling failures for difficult buildings, but rather reflects the exploration of different regions in the model space—some more suitable than others for a given building's thermal characteristics.

By treating model generation as a stochastic sampling process and using validation-based selection, we effectively transform the inherent variability of LLM outputs from a source of uncertainty into a mechanism for robust model discovery. This approach is analogous to ensemble methods in machine learning, where diversity among models contributes to improved generalization, though here the diversity emerges naturally from the LLM's generative process rather than from algorithmic design choices.

4.6 Cost Analysis

Having demonstrated our framework's performance under varying design choices in previous sections, we now analyze their cost implications. Specifically, we examine the costs by measuring token consumption during the LLM-assisted design phase, comparing token usage separately for physics-based and neural models. In Table 1, we present token usage for several design choices (reported as the 90th or 50th percentile). The statistics for token usage are calculated on a per-home and per-experiment basis; for example, P90 indicates that in 90% of homes, the token usage for a given experiment was at or below the value reported in the table.

We present several design choices in the table. The default configuration, shown in the first row as "Workflow with basic prompt", uses a basic prompt for physics-based modeling, a full architecture search prompt for neural modeling, and has agent control disabled ($\delta = 0$). In the second design choice, shown as "Workflow with physics modeling knowledge in prompt", we change the physics-based model generation prompt to include expert knowledge. Finally, in the last experiment shown as "Agent control ($\delta = 1$) with basic prompt", we enable the agent to control both the transition from physics-based to neural modeling and when to complete neural model design (i.e., $\delta = 1$).

The token usage analysis reveals distinct patterns between physics-based and neural model design processes. Physics-based model design consistently requires significantly higher token consumption compared to neural model design. This disparity can be attributed to several factors. First, in our experiments we use a larger number of generations for physics-based models (10) compared to neural models (5). Additionally, physics-based model generations experience higher failure rates, necessitating additional attempts to achieve the target number of successful generations. These failures compound token usage as information from previous generations is incorporated into subsequent prompts to improve success rates.

When examining agent control in physics-based model design, we observe notably lower token consumption. This reduction results from fewer generations when the agent autonomously makes transition decisions. As discussed previously, this comes at the cost of performance when considering only a single run. However, if we apply the stochasticity-based model selection introduced in Section 4.5, we can achieve similar results at 14% of the cost compared to the workflow approach.

5 Conclusion

This work presents the first empirical evidence of the effectiveness of using LLM-powered approaches for automating building thermal dynamics modeling. To our knowledge, it is also the first large-scale demonstration of these approaches on real-world datasets of dynamical systems. In addition to introducing a flexible framework for hybrid neuro-physical modeling that can be used by the community with different prompts, language models and/or datasets, our extensive tests on data from 300 homes yield several key conclusions and recommendations for practitioners and researchers alike.

While our framework is intended to serve as a tool for exploring this new LLM-powered modeling paradigm, our results indicate that with minimal tuning the resulting models perform on par with automated thermal dynamic modeling approaches proposed to date, achieving RMSE values lower than ~2 Fahrenheit in over 95% of the 24-hour predictions that were made across all sensors and homes in our dataset.

Physics Modeling: First, and echoing findings from prior work on introducing expert knowledge/guidance into prompts [8], we found that our expert-guided prompt did not provide significant difference in modeling accuracy when compared to a prompt without expert knowledge. Thus, our recommendation for practitioners trying out ThermalForge is to keep the prompt simple and focused on describing the data and modeling goals, similar to our non-expert prompt. As for researchers wishing to improve upon our framework, we have a few more recommendations. Exploring the space of all possible prompts is impossible, but it is reasonable to expect that better prompting strategies could lead to better results. For instance, we did not attempt to make use of auxiliary information in our models that could help explain variability in terms of proxies for solar irradiance, occupancy patterns, etc. Similarly, providing LLM agents with access to tools designed to create and calibrate white and/or grey-box models of thermal dynamics (e.g.,

Table 1: Token usage per experiment / home

Design choice	Tokens			
	Physics		Neural	
	P90	P50	P90	P50
Workflow with basic prompt	623,049	424,659	88,314	76,467
Workflow with physics modeling knowledge in prompt	634,562	411,725	87,247	75,161
Agent control ($\delta = 1$) with basic prompt	86,014	53,497	96,674	57,982

[22, 37]) could result in more accurate models though this remains to be tested. Nevertheless, the fact that the prompt P_{pg} containing minimal expert guidance performed well in our experiments is a testament to the amount of knowledge already encoded in modern LLMs. We would have also liked to test the validity of the resulting parameters for the generated physics-based models. However, the candidate models f_{θ_p} did not always strictly adhere to the linear RC formulation resulting in parameter sets Θ that were not comparable across generations.

Neural Modeling: We found that the LLM required very little guidance to generate g_{θ_p} candidates that were adept at absorbing the difference between the physics-based roll-out of the dynamics and the observed/measured values. In practice, we recommend allowing the LLM agent to perform neural architecture search unless there is a strong reason to believe that a particular architecture is preferable, in which case limiting the optimization to just optimal parameter selection would be more appropriate. As for future research efforts, we suggest exploring other neural architectures beyond U-Net, and a more careful analysis of the physics-based roll-out residuals $\epsilon_{n:n+D}$ to better understand the neural modeling task and its input data distribution.

Convergence check: For single runs of ThermalForge, our experiments clearly suggest that agentic stopping $\delta = 1$ is counterproductive using the prompt P_{na} we designed (at least for the physics based modeling phase). However, given the results of our experiments in the stochasticity analysis, which showed virtually the same performance for $\delta = 1$ and $\delta = 0$ when selecting the best out of multiple runs, we recommend setting $\delta = 1$ since this one has a lower cost (in terms of token usage).

Cost analysis: We found that the majority of our token budget was spent *searching* for the right physics-based model, even though the neural model was able to absorb most of the resulting variance in all cases. We thus recommend that practitioners carefully consider how much they value model interpretability as the costs of searching through the physics-based function space can be costly. Similarly, capitalizing on the stochastic results of each run we showed how selecting the best model of multiple runs can reduce token usage (with $\delta = 1$) without significantly affecting model performance. In terms of future research, it is clear that finding more efficient physics-based modeling strategies (tool usage, better prompting, etc.) may reduce the costs of model generation.

Other considerations: While we evaluated a number of different design choices for the framework we acknowledge that there is more to explore. Our hope is that by releasing our ThermalForge implementation to the community, others will continue to produce results that can inform our path forward and get us closer to truly scalable automatic thermal dynamics modeling for all buildings. For example, future work might consider: experimenting with more feature-rich agent harnesses including access to thermal modeling software toolboxes; studying the physical grounding of the estimated parameters θ_p; testing the performance across different prediction horizons (besides 24 hours) or data resolutions (besides hourly); and studying the effect of input and output temporal window sizes. Moreover, there are potential modifications to ThermalForge that would allow exploration of additional design choices not considered in our study. For instance, future work might explore different compositions of the physics (f_{θ_p}) and neural (g_{θ_n}) functions beyond the residual of the rolled-out physics model and/or allowing the LLM agent to modify these composition operators. Similarly, there could be value in exploring other ways to aggregate samples from the stochastic solutions to the bi-level optimization problem. The best-of-N strategy we followed should be compared against others [11].

Lastly, we want to reiterate that though this framework was designed, applied and tested to study agentic modeling frameworks in the context of thermal dynamics in buildings, it is in principle applicable to any other dynamical system. We plan to extend the application to other systems and use cases in future work.

References

[1] Clarence Agbi, Zhen Song, and Bruce Krogh. 2012. Parameter identifiability for multi-zone building models. In *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*. 6951–6956. doi:10.1109/CDC.2012.6425995

[2] Klaus Kaae Andersen, Henrik Madsen, and Lars H Hansen. 2000. Modelling the heat dynamics of a building using stochastic differential equations. *Energy and Buildings* 31, 1 (2000), 13–24.

[3] Krzysztof Arendt, Muhyiddine Jradi, Hamid Reza Shaker, and Christian Veje. 2018. Comparative analysis of white-, gray-and black-box models for thermal simulation of indoor environment: Teaching building case study. In *Building Performance Analysis Conference and SimBuild: Co-organized by ASHRAE and IBPSA-USA*. ASHRAE, 173–180.

[4] Ercan Atam and Lieve Helsen. 2016. Control-Oriented Thermal Modeling of Multizone Buildings: Methods and Issues: Intelligent Control of a Building System. *IEEE Control Systems Magazine* 36, 3 (2016), 86–111. doi:10.1109/MCS.2016.2535913

[5] David H Blum and Michael Wetter. 2017. MPCPy: An open-source software platform for model predictive control in buildings. In *Building Simulation 2017*, Vol. 15. IBPSA, 1381–1390.

[6] Kathleen Champion, Bethany Lusch, J Nathan Kutz, and Steven L Brunton. 2019. Data-driven discovery of coordinates and governing equations. *Proceedings of the National Academy of Sciences* 116, 45 (2019), 22445–22451.

[7] Gaurav Chaudhary, Hicham Johra, Laurent Georges, and Bjørn Austbø. 2025. Transfer learning in building dynamics prediction. *Energy and Buildings* 330 (2025), 115384. doi:10.1016/j.enbuild.2025.115384

[8] Ziru Chen, Shijie Chen, Yuting Ning, Qianheng Zhang, Boshi Wang, Botao Yu, Yifei Li, Zeyi Liao, Chen Wei, Zitong Lu, et al. 2024. Scienceagentbench: Toward rigorous assessment of language agents for data-driven scientific discovery. *arXiv preprint arXiv:2410.05080* (2024).

[9] Drury B. Crawley, Linda K. Lawrie, Frederick C. Winkelmann, W.F. Buhl, Y.Joe Huang, Curtis O. Pedersen, Richard K. Strand, Richard J. Liesen, Daniel E. Fisher, Michael J. Witte, and Jason Glazer. 2001. EnergyPlus: creating a new-generation

building energy simulation program. *Energy and Buildings* 33, 4 (2001), 319–331. doi:10.1016/S0378-7788(00)00114-6 Special Issue: BUILDING SIMULATION'99.

[10] Ján Drgoňa, Aaron R. Tuor, Vikas Chandan, and Draguna L. Vrabie. 2021. Physics-constrained deep learning of multi-zone building thermal dynamics. *Energy and Buildings* 243 (2021), 110992. doi:10.1016/j.enbuild.2021.110992

[11] Weihua Du, Yiming Yang, and Sean Welleck. 2025. Optimizing Temperature for Language Models with Multi-Sample Inference. *arXiv preprint arXiv:2502.05234* (2025).

[12] ecobee. 2025. Donate Your Data. Retrieved September 7, 2025 from https://www.ecobee.com/en-us/donate-your-data/

[13] Zhen Fang, Nicolas Crimier, Lisa Scanu, Alphanie Midelet, Amr Alyafi, and Benoit Delinchant. 2021. Multi-zone indoor temperature prediction with LSTM-based sequence to sequence model. *Energy and Buildings* 245 (2021), 111053. doi:10.1016/j.enbuild.2021.111053

[14] Pietro Favaro, Jean-François Toubeau, François Vallée, and Yury Dvorkin. 2025. Decision-Focused Learning for Complex System Identification: HVAC Management System Application. In *Proceedings of the 16th ACM International Conference on Future and Sustainable Energy Systems (E-Energy '25)*. Association for Computing Machinery, New York, NY, USA, 347–358. doi:10.1145/3679240.3734584

[15] Mingquan Feng, Yixin Huang, Yizhou Liu, Bofang Jiang, and Junchi Yan. 2025. PhysPDE: Rethinking PDE discovery and a physical HYpothesis selection benchmark. In *The Thirteenth International Conference on Learning Representations*.

[16] Francesco Ferracuti, Alessandro Fonti, Lucio Ciabattoni, Stefano Pizzuti, Alessia Arteconi, Lieve Helsen, and Gabriele Comodi. 2017. Data-driven models for short-term thermal behaviour prediction in real buildings. *Applied Energy* 204 (2017), 1375–1387. doi:10.1016/j.apenergy.2017.05.015

[17] Siddhartha Ghosh, Steve Reece, Alex Rogers, Stephen Roberts, Areej Malibari, and Nicholas R. Jennings. 2015. Modeling the Thermal Dynamics of Buildings: A Latent-Force- Model-Based Approach. *ACM Trans. Intell. Syst. Technol.* 6, 1, Article 7 (March 2015), 27 pages. doi:10.1145/2629674

[18] Samuel Holt, Tennison Liu, and Mihaela van der Schaar. 2024. Automatically learning hybrid digital twins of dynamical systems. *Advances in Neural Information Processing Systems* 37 (2024), 72170–72218.

[19] Brent Huchuk, Scott Sanner, and William O'Brien. 2022. Evaluation of data-driven thermal models for multi-hour predictions using residential smart thermostat data. *Journal of Building Performance Simulation* 15, 4 (2022), 445–464.

[20] Zhanhong Jiang and Young M Lee. 2019. Deep transfer learning for thermal dynamics modeling in smart buildings. In *2019 IEEE international conference on big data (Big data)*. IEEE, 2033–2037.

[21] Kevin J. Kircher and K. Max Zhang. 2015. On the lumped capacitance approximation accuracy in RC network building models. *Energy and Buildings* 108 (2015), 454–462. doi:10.1016/j.enbuild.2015.09.053

[22] Kevin J Kircher and K Max Zhang. 2016. Testing building controls with the BLDG toolbox. In *2016 American Control Conference (ACC)*. IEEE, 1472–1477.

[23] Justin Koeln, Bryan Keating, Andrew Alleyne, Christopher Price, and Bryan P Rasmussen. 2017. Multi-zone temperature modeling and control. In *Intelligent Building Control Systems: A Survey of Modern Building Control and Sensing Strategies*. Springer, 139–166.

[24] Julien Leprince, Henrik Madsen, Clayton Miller, Jaume Palmer Real, Rik van der Vlist, Kaustav Basu, and Wim Zeiler. 2022. Fifty shades of grey: Automated stochastic model identification of building heat dynamics. *Energy and Buildings* 266 (2022), 112095. doi:10.1016/j.enbuild.2022.112095

[25] Han Li, William O'Brien, Vivian Loftness, Erica Cochran Hameen, and Tianzhen Hong. 2025. A critical review of use cases and insights from a large dataset of smart thermostats. *Advances in Applied Energy* 19 (2025), 100236. doi:10.1016/j.adapen.2025.100236

[26] Han Li, Giuseppe Pinto, Marco Savino Piscitelli, Alfonso Capozzoli, and Tianzhen Hong. 2024. Building thermal dynamics modeling with deep transfer learning using a large residential smart thermostat dataset. *Engineering Applications of Artificial Intelligence* 130 (2024), 107701. doi:10.1016/j.engappai.2023.107701

[27] Na Luo and Tianzhen Hong. 2022. *Ecobee donate your data 1,000 homes in 2017*. Technical Report. Pacific Northwest National Lab.(PNNL), Richland, WA (United States).

[28] Pingchuan Ma, Tsun-Hsuan Wang, Minghao Guo, Zhiqing Sun, Joshua B. Tenenbaum, Daniela Rus, Chuang Gan, and Wojciech Matusik. 2024. LLM and Simulation as Bilevel Optimizers: A New Paradigm to Advance Physical Scientific Discovery. In *Proceedings of the 41st International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 235)*, Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (Eds.). PMLR, 33940–33962. https://proceedings.mlr.press/v235/ma24m.html

[29] Francesco Massa Gray and Michael Schmidt. 2018. A hybrid approach to thermal building modelling using a combination of Gaussian processes and grey-box models. *Energy and Buildings* 165 (2018), 56–63. doi:10.1016/j.enbuild.2018.01.039

[30] Clémence Métayer, Annabelle Ballesta, and Julien Martinelli. 2025. Data-driven Discovery of Digital Twins in Biomedical Research. *arXiv preprint arXiv:2508.21484* (2025).

[31] Tung Nguyen, Arsh Koneru, Shufan Li, and Aditya Grover. 2025. PhysiX: A Foundation Model for Physics Simulations. arXiv:2506.17774 [cs.LG] https://arxiv.org/abs/2506.17774

[32] U.S. Department of Energy. 2025. EnergyPlus. Retrieved September 7, 2025 from https://energyplus.net/

[33] Mathias Perslev, Michael Jensen, Sune Darkner, Poul Jø rgen Jennum, and Christian Igel. 2019. U-Time: A Fully Convolutional Network for Time Series Segmentation Applied to Sleep Staging. In *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Eds.). Curran Associates, Inc., 4415–4426.

[34] Giuseppe Pinto, Riccardo Messina, Han Li, Tianzhen Hong, Marco Savino Piscitelli, and Alfonso Capozzoli. 2022. Sharing is caring: An extensive analysis of parameter-based transfer learning for the prediction of building thermal dynamics. *Energy and Buildings* 276 (2022), 112530. doi:10.1016/j.enbuild.2022.112530

[35] Fabian Raisch, Thomas Krug, Christoph Goebel, and Benjamin Tischler. 2025. GenTL: A General Transfer Learning Model for Building Thermal Dynamics. In *Proceedings of the 16th ACM International Conference on Future and Sustainable Energy Systems (E-Energy '25)*. Association for Computing Machinery, New York, NY, USA, 322–333. doi:10.1145/3679240.3734589

[36] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. 2015. U-Net: Convolutional Networks for Biomedical Image Segmentation. *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015* (2015), 234–241.

[37] Elliott Skomski, Cameron Rutherford, Aaron Tuor, Jan Drgona, Elliott Skomski, Soumya Vasisht, and Draguna Vrabie. 2021. pnnl/neuromancer. [Computer Software] https://doi.org/10.11578/dc.20240614.181. doi:10.11578/dc.20240614.181

[38] Thermal Energy System Specialists. 2025. TRNSYS: Transient System Simulation Tool. Retrieved September 7, 2025 from https://www.trnsys.com/

[39] Daniel Stoller, Sebastian Ewert, and Simon Dixon. 2018. Wave-U-Net: A Multi-Scale Neural Network for End-to-End Audio Source Separation. In *Proceedings of the 19th International Society for Music Information Retrieval Conference, ISMIR 2018, Paris, France, September 23-27, 2018*, Emilia Gómez, Xiao Hu, Eric Humphrey, and Emmanouil Benetos (Eds.). 334–340.

[40] Marius Tacke, Matthias Busch, Kartik Bali, Kian Abdolazizi, Kevin Linka, Christian Cyron, and Roland Aydin. 2025. Constitutive scientific generative agent (CSGA): Leveraging large language models for automated constitutive model discovery. *Machine Learning for Computational Science and Engineering* 1, 1 (2025), 23.

[41] Charalampos Vallianos, Andreas Athienitis, and Benoit Delcroix. 2022. Automatic generation of multi-zone RC models using smart thermostat data from homes. *Energy and Buildings* 277 (2022), 112571. doi:10.1016/j.enbuild.2022.112571

[42] Charalampos Vallianos, José Candanedo, and Andreas Athienitis. 2023. Application of a large smart thermostat dataset for model calibration and Model Predictive Control implementation in the residential sector. *Energy* 278 (2023), 127839. doi:10.1016/j.energy.2023.127839

[43] Charalampos Vallianos, José Candanedo, and Andreas Athienitis. 2024. Thermal modeling for control applications of 60,000 homes in North America using smart thermostat data. *Energy and Buildings* 303 (2024), 113811. doi:10.1016/j.enbuild.2023.113811

[44] Florian Wiesner, Matthias Wessling, and Stephen Baek. 2025. Towards a Physics Foundation Model. arXiv:2509.13805 [cs.LG] https://arxiv.org/abs/2509.13805

[45] Ziyao Yang, Amol D. Gaidhane, Ján Drgoňa, Vikas Chandan, Mahantesh M. Halappanavar, Frank Liu, and Yu Cao. 2024. Physics-constrained graph modeling for building thermal dynamics. *Energy and AI* 16 (2024), 100346. doi:10.1016/j.egyai.2024.100346

[46] Danilo Yu, Abdolreza Abhari, Alan S. Fung, Kaamran Raahemifar, and Farahnaz Mohammadi. 2018. Predicting indoor temperature from smart thermostat and weather forecast data. In *Proceedings of the Communications and Networking Symposium* (Baltimore, Maryland) *(CNS '18)*. Society for Computer Simulation International, San Diego, CA, USA, Article 9, 12 pages.