BuilDyn: Excitation-Driven Data Generation for Building Thermal Dynamics Modeling and Control

Felix Koch*
Technical University of Applied Sciences Rosenheim
Rosenheim, Bavaria, Germany
felix.koch@th-rosenheim.de

Thomas Krug*
Technical University of Applied Sciences Rosenheim
Rosenheim, Bavaria, Germany
thomas.krug@th-rosenheim.de

Fabian Raisch**
Technical University of Applied Sciences Rosenheim
Rosenheim, Bavaria, Germany
fabian.raisch@th-rosenheim.de

Benjamin Schäfer
Karlsruhe Institute of Technology
Karlsruhe, Germany
benjamin.schaefer@kit.edu

Benjamin Tischler
Technical University of Applied Sciences Rosenheim
Rosenheim, Bavaria, Germany
benjamin.tischler@th-rosenheim.de

Abstract

Machine learning (ML) is increasingly used for data-driven modeling of buildings to enable downstream tasks such as fault detection and diagnosis, and energy-efficient control. While recent work improves generalization across building characteristics, weather, and occupancy, generalization also depends on sufficient exploration of the control-driven system state space. Existing real-world datasets and simulation environments predominantly reflect stationary operation under fixed control policies, resulting in limited excitation and reduced robustness to unseen operating conditions.

This paper introduces BuilDyn, a package based on BuilDa that enables customizable excitation strategies for control-oriented data generation. BuilDyn further supports sampling from representative building distributions and provides a Python interface for easy integration into machine learning pipelines. We demonstrate the benefits of BuilDyn by comparing the performance of data-driven ML models trained on non-excited and excited data for one building. With BuilDyn, we hope to advance scalable control-oriented modeling and support future directions such as transfer learning and building-specific foundation models.

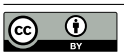
CCS Concepts

• **Software and its engineering** → **Software libraries and repositories**; • **Computing methodologies** → *Simulation support systems.*

Keywords

software, simulation, building thermal dynamics, excitation

*Also with Karlsruhe Institute of Technology.
**Main affiliation at Technical University of Applied Sciences Rosenheim; doctoral candidate at Technical University of Munich (cooperative doctorate with Technical University of Applied Sciences Rosenheim).



ACM Sustainability Week Companion '26, Banff, AB, Canada

ACM ISBN 979-8-4007-2199-1/26/06
https://doi.org/10.1145/3765611.3815432

ACM Reference Format:
Felix Koch, Thomas Krug, Fabian Raisch, Benjamin Schäfer, and Benjamin Tischler. 2026. BuilDyn: Excitation-Driven Data Generation for Building Thermal Dynamics Modeling and Control. In *ACM Sustainability Week 2026 (ACM Sustainability Week Companion '26), June 22–25, 2026, Banff, AB, Canada.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3765611.3815432

1 Introduction

Building operations account for roughly 30% of global energy consumption [10]. Consequently, reducing energy demand through fault detection and diagnosis (FDD), as well as energy-efficient control, applications due to its scalability across diverse buildings and its ability to learn complex system behavior from data. In several studies, Machine Learning (ML) has demonstrated superior performance to conventional methods [5, 21]. ML-based modeling has demonstrated the option of generalization through transfer learning, thereby addressing data availability as in [22, 24, 25], and has been shown to have the best performance in continuous learning frameworks [26]. However, ML methods have a key limitation: poor performance on out-of-distribution data [7]. This makes data generated through state-space exploration essential, as models trained on insufficiently excited operating conditions fail to generalize to unseen system dynamics and control regimes. Excitation denotes the deliberate stimulation of control inputs to actively explore the state-space.

A similar challenge appears in conventional data-driven modeling for buildings based on Resistance-Capacitance (RC) gray-box models, where low-quality data leads to poor parameter estimation [28]. To address this issue, studies employ excitation strategies to enhance data richness and improve model accuracy. For example, Madsen and Schultz [2, 20] proposed a pseudo-random binary sequence (PRBS) with both short and long time constants to capture air and envelope dynamics. Other works similarly use tailored excitation signals to improve parameter estimation [1, 14, 27]. However, manipulating the operation of a building to collect high-quality data is often not possible in real buildings due to occupancy constraints [9].

An alternative for both conventional data-driven and ML-based approaches is the use of simulated datasets. For example, [27] provides data from six real-world buildings, including dedicated excitation periods that enable efficient ML-based modeling and control, as demonstrated in [32]. However, such datasets are typically designed for specific use cases and have restricted large-scale applicability. Large-scale datasets of real-world or simulated buildings, such as [3, 19, 23], partially address this limitation. Such datasets are particularly valuable for transfer learning applications, which use large-scale datasets to train ML models that can be later fine-tuned to real buildings with limited data [8]Nevertheless, these large-scale datasets predominantly reflect operations constrained by existing control policies and therefore lack sufficient excitation, limiting their suitability for learning robust models for FDD and control.

A third option is to use data generators or dedicated simulation environments. These range from complex white-box models based on EnergyPlus [6], Modelica [31], or TRNSYS [13] to more user-friendly environments that simplify control and benchmarking, such as Sinergym [11] or BoPTEST [4]. White-box models offer the benefit of a complete physical record of the building dynamics, but they are computationally expensive and require significant expertise to set up and operate. More user-friendly environments provide easier access to building simulations, but often rely on a limited number of fixed building models and other constraints, which may not capture the diversity of real-world buildings. Krug et al. [16] combine the benefits of high flexibility and ease of use by providing an FMU-based simulation environment that can be customized to different building envelopes, locations, and occupancy profiles. So far, this tool lacks dedicated excitation strategies that enable control-oriented modeling.

To address this gap, we investigate excitations for buildings to generate more robust data for training ML-based building models using three different excitation strategies. We show how such strategies explore simulated building state-spaces more thoroughly than in controlled buildings. Based on these excitation strategies, we show that data generated with such excitations can improve the predictive accuracy of ML models across a wider range of state spaces and dynamics. Additionally, we introduce the Python package *BuilDyn*. The package allows for the implementation of custom excitation strategies for building simulations based on BuilDa [16] to generate rich datasets to train and evaluate ML models for control. *BuilDyn* also extends the original framework by enabling sampling from representative distributions of building domains (e.g., buildings constructed between 1980 and 2000 in a specific country), combined with functionalities for easier use.

The remainder of this paper is structured as follows. Section 2 introduces *BuilDyn*. Section 3 evaluates excitation strategies from *BuilDyn* on a building modeling task and compares excited and non-excited data. Finally, Section 4 concludes the paper and discusses future applications.

2 BuilDyn

In this section, we introduce *BuilDyn*, a Python package that extends BuilDa [16]. BuilDa provides a framework for generating large-scale, realistic building data, while *BuilDyn* adds functionality to dynamically manipulate control inputs and generate excited trajectories for control-oriented modeling. For an in-depth description of BuilDa and its configurable FMU, we refer to [16]; additional details on the relationship between BuilDa and *BuilDyn* are provided in the appendix.

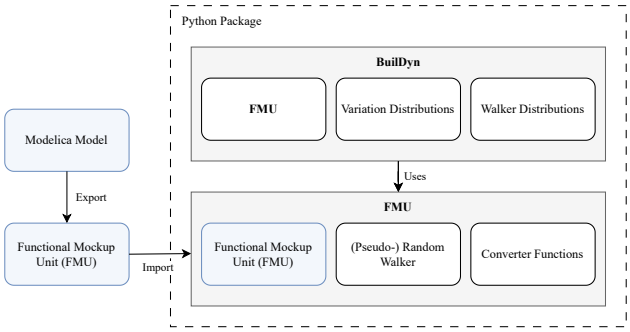


Figure 1: Structure of the *BuilDyn* Python package.

2.1 Architecture

The *BuilDyn* package consists of two main components visualized in Figure 1: The FMU and the BuilDyn class. The FMU class is a wrapper for the FMU API FmPy [29], providing an interface to simulate buildings. In *BuilDyn*, the FMU of BuilDa is slotted in. However, *BuilDyn* is also designed to hold other FMUs. The BuilDyn class, on the other hand, is a wrapper for the FMU class that enables the creation of FMU objects and FMU simulations from user-defined distributions. *BuilDyn* and further tutorials are also available on GitHub.

2.2 Key Concepts

BuilDyn was designed incrementally on BuilDa around the following four key concepts/features.

F1: Excitations. Excitations, such as pseudo-random inputs or stochastic signals, are commonly used in system identification to explore different states of a system. *BuilDyn* introduces the concept of walkers, which enable the user to define excitations for the dynamic system that trigger in configurable time intervals. These walkers enable more thorough state explorations of buildings compared to standard control methods. In particular, three complementary excitation strategies are employed: (i) a *Poisson excitation*, which applies piecewise-constant heating levels for randomly sampled durations to create irregular step-like inputs; (ii) a *sinusoidal excitation*, which chains sinusoidal segments with varying frequency and amplitude separated by short steady-state holds to mimic diverse periodic heating patterns; and (iii) a *ramp excitation*, which repeatedly ramps linearly between minimum and maximum values with holds at the extrema to systematically probe transitional dynamics. Together, these walkers provide a mix of stochastic persistence, periodic variation, and structured transitions, improving coverage of the control-dependent state-space. For further details on the three strategies, see Appendix B.

F2: User-Friendly FMUs. *BuilDyn* provides an FMU class that serves as an abstraction layer to the FmPy [29] API. This abstraction

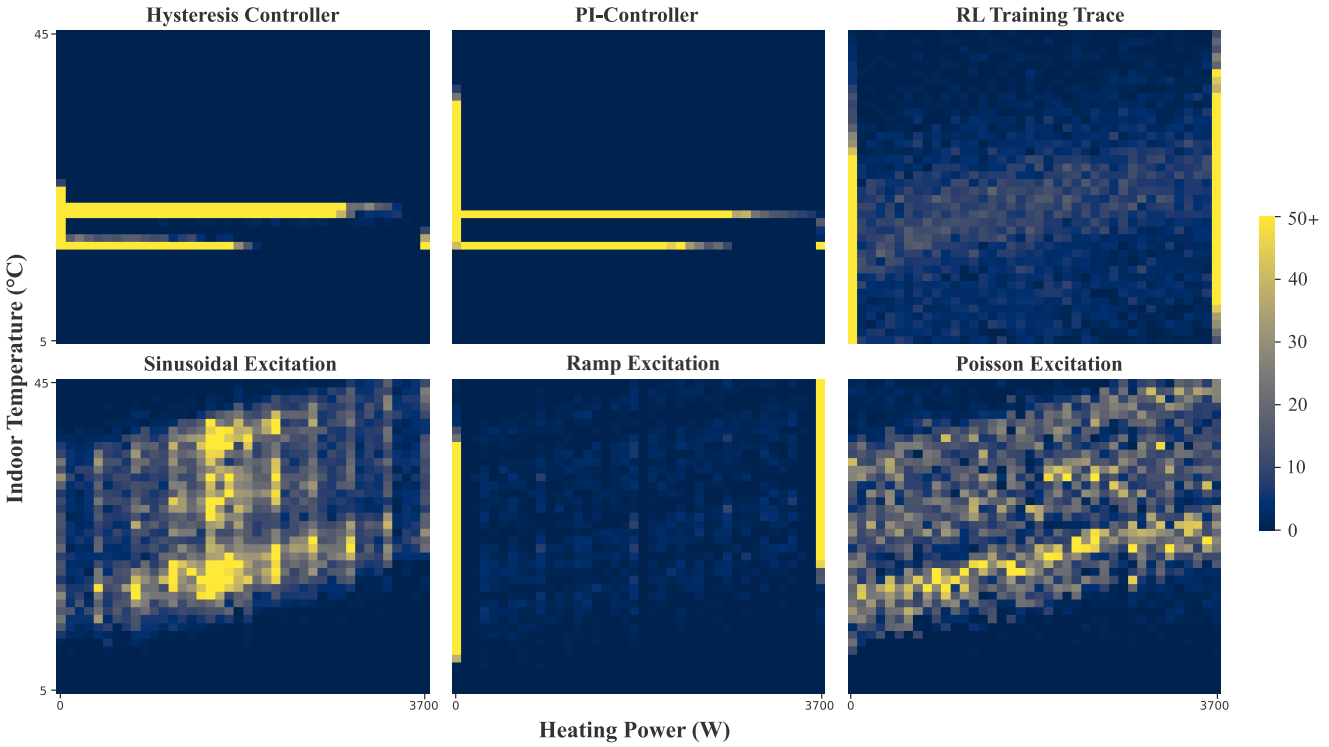


Figure 2: State-Exploration of BuilDyn with different excitation/control strategies

provides users with a tool to simulate buildings and manipulate their envelope parameters, weather scenarios, and occupant profiles. Furthermore, *BuilDyn*, as a Python package, is distributed on PyPI, enabling seamless integration of FMU functionality into pipelines that require building simulations.

F3: Custom Variation Distributions. Real-world building populations exhibit different distributions of envelope parameters and characteristics. With *BuilDyn*, we enable users to custom define those distributions over building parameters to sample from. This enables users to reconstruct real-world building envelope distributions (for instance, centered on TABULA [18] archetypes) in their benchmarks.

F4: Converter Functions. Building parameters may require successive recalculations when modified. For instance, cooling-/heating-loads are dependent on the volume of a building. To allow users to define such chains of dependent updates, *BuilDyn* adopts the idea of the converter layer functionality from [16] but offers users full control over conversions through a dedicated Python interface.

3 Experiments

This section illustrates two experiments on different excitation strategies using the BuilDyn package. In Section 3.1, we compare different control and exploration strategies with respect to state-space coverage. Subsequently, Section 3.2 employs two of these strategies to generate data, which is then used to train an ML prediction model in the next step.

3.1 State-exploration with excitations

In this experiment, we compare the three excitation strategies implemented in BuilDyn (see Section 2.2) with two standard control-based data generations and an RL training process. For standard controls, we use a PI and a hysteresis controller, as these are typically used in reality. For demonstration, we use the standard pre-implemented building by [16] in a heat control scenario. Figure 2 presents the resulting state-space coverage by counting and binning the appearing combinations of indoor temperature and heating signal for each 15-minute timestep over 1 year. The results demonstrate that the PI and hysteresis controllers only explore a narrow state band, as the system is only explored by driving disturbances, i.e., outdoor conditions and occupancy. The RL agent, on the contrary, explores a larger state space as part of its learning process. Consequently, ML models trained on data generated by these strategies more likely need to extrapolate when deployed in downstream tasks such as MPC- or RL-based control. In contrast, the three *BuilDyn* excitation strategies exhibit substantially greater overlap with the RL trace, suggesting their superior suitability as training data for ML-based building models intended for control. In particular, the different excitation strategies cover different parts of the TL training trace in this concrete example.

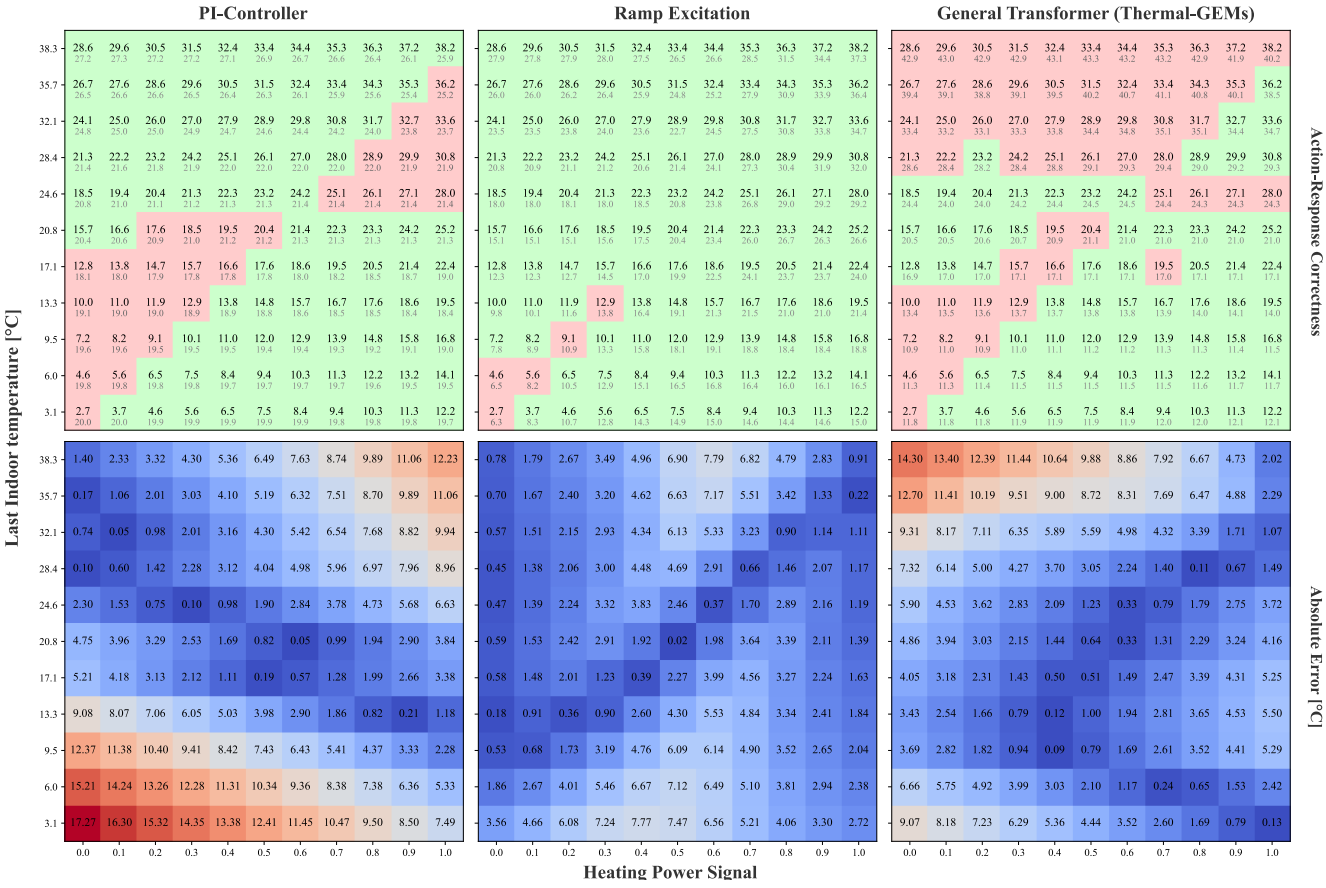


Figure 3: Top row: ARC of ML-Models with differently excited/controlled training data. Green background indicates correct response direction. The larger cell fonts show the ground truth, and the grey numbers show the predicted indoor temperature. Bottom row: Shows AE values with a better fit indicated by a blue color.

3.2 Excitation effects on ML models

In this section, two of the strategies from Section 3.1 are employed to develop an ML–based prediction model for building thermal dynamics. The first model is trained on data generated by the PI-controller simulation with a heating setpoint of 22 °C, similar to [26]. The second model uses data originating from BuilDyn using the ramp excitation. In both cases, a timestep of 900 s (15 min) is used for a 1-year simulation (January to December). For training, we use a transformer model [30], following [15]. Additionally, we compare a third model from [15] that is trained on multiple controlled source buildings, which provides generalization capabilities across locations, occupancies, and building properties.

All three models are evaluated on February 1 of the second year. To obtain diverse initial thermal conditions, the simulation is first run throughout January of the second year, resulting in different indoor temperatures at 00:00 on February 1st[1]. From this initialized state, a set of discrete control actions ranging from 0 (off) to 1 (full power) is applied. For each action, the resulting indoor temperature at the next time step (00:15) is recorded and compared to the corresponding prediction of the ML model. Performance is assessed using the *absolute error* (AE) between prediction model and ground truth as well as the *action-response correctness* (ARC). The ARC evaluates whether the model correctly captures the physical direction of the temperature change.

Figure 3 illustrates model behaviors across the different training strategies. Results show more green entries in the first row and blue entries in the second row for the excited scenario. The model trained on data generated within a PI-controlled building achieved 76% ARC, while the model trained on data from buildings with ramp excitations achieved a 96% ARC, which is the highest among all excitation strategies applied (complementary results are illustrated in Figure 7). The generalized transformer model we employed, based on [15], only achieved a 54% ARC, further motivating the use of data generated in excited building simulations for generalized data-driven models. This indicates that excitation-based modeling improves robustness to unseen states, yielding lower absolute errors and more accurate control responses. In contrast, models trained without excitation exhibit pronounced extrapolation errors, often reverting to values typical of the training distribution.

[1]At 00:00, the outdoor temperature was 0.0°C, and there was no solar radiation.

4 Conclusion

We present *BuilDyn*, an open-source Python package for simulating and exciting buildings modeled with FMUs. Using an example building simulated in *BuilDyn*, we highlight the necessity of excitation in data generation for ML-based building modeling. We show how excitations defined in *BuilDyn* can help mitigate extrapolation behavior in ML-based models for buildings. Limitations of this work include the extrapolation behavior of ML-based building models, which has only been investigated superficially. Seasonal effects and other covariates beyond the control signal and indoor temperature have not been considered. Although we observed similar patterns when evaluating the behavior of ML models at other times and in other seasons, broader evaluations are required. As transfer learning is an efficient method for modeling building thermal dynamics [25], future work may focus on excitation strategies to improve the robustness of generalized models under extrapolation. *BuilDyn* is envisioned as a community tool for benchmarking, manipulating, and exciting buildings, thereby contributing to the development of more robust generalized building models.

References

[1] Peder Bacher and Henrik Madsen. 2010. *Experiments and Data for Building Energy Performance Analysis: Financed by The Danish Electricity Saving Trust.* Technical University of Denmark, DTU Informatics, Building 321.

[2] Peder Bacher and Henrik Madsen. 2011. Identifying suitable models for the heat dynamics of buildings. *Energy and Buildings* 43, 7 (July 2011), 1511–1522. doi:10.1016/j.enbuild.2011.02.005

[3] Anaïs Berkes, Yoshua Bengio, David Rolnick, and Donna Vakalis. 2025. A HOT Dataset: 150,000 Buildings for HVAC Operations Transfer Research. In *Proceedings of the 12th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation.* 171–180.

[4] David Blum, Javier Arroyo, Sen Huang, Ján Drgoňa, Filip Jorissen, Harald Taxt Walnum, Yan Chen, Kyle Benne, Draguna Vrabie, Michael Wetter, et al. 2021. Building optimization testing framework (BOPTEST) for simulation-based benchmarking of control strategies in buildings. *Journal of Building Performance Simulation* 14, 5 (2021), 586–610.

[5] Wonjun Choi and Sangwon Lee. 2023. Performance evaluation of deep learning architectures for load and temperature forecasting under dataset size constraints and seasonality. *Energy and buildings* 288 (2023), 113027.

[6] Drury B Crawley, Linda K Lawrie, Frederick C Winkelmann, Walter F Buhl, Y Joe Huang, Curtis O Pedersen, Richard K Strand, Richard J Liesen, Daniel E Fisher, Michael J Witte, et al. 2001. EnergyPlus: creating a new-generation building energy simulation program. *Energy and buildings* 33, 4 (2001), 319–331.

[7] Jan Marco Ruiz de Vargas, Fabian Raisch, Zoltan Nagy, Pierre Pinson, and Christoph Goebel. 2026. Counter-Dyna: Data-Efficient RL-Based HVAC Control using Counterfactual Building Models. *arXiv preprint* (2026).

[8] Hongwen Dou and Kun Zhang. 2025. Transfer learning for cross-building forecasting of building energy and indoor air temperature in model predictive control applications. *Journal of Building Engineering* 111 (2025), 113341.

[9] Hassan Harb, Neven Boyanov, Luis Hernandez, Rita Streblow, and Dirk Müller. 2016. Development and validation of grey-box models for forecasting the thermal response of occupied buildings. *Energy and Buildings* 117 (2016), 199–207. doi:10.1016/j.enbuild.2016.02.021

[10] IEA. 2023. *Tracking Clean Energy Progress 2023.* Technical Report. International Energy Agency. https://www.iea.org/reports/tracking-clean-energy-progress-2023

[11] Javier Jiménez-Raboso, Alejandro Campoy-Nieves, Antonio Manjavacas-Lucas, Juan Gómez-Romero, and Miguel Molina-Solana. 2021. Sinergym: a building simulation and control framework for training reinforcement learning agents. In *Proceedings of the 8th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation* (Coimbra, Portugal) *(BuildSys '21).* Association for Computing Machinery, New York, NY, USA, 319–323. doi:10.1145/3486611.3488729

[12] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[13] S. A. Klein, W. A. Beckman, J. W. Mitchell, J. A. Duffie, T. L. Freeman, J. C. Mitchell, J. E. Braun, B. L. Evans, J. P. Kummer, R. E. Urban, A. Fiksel, J. W. Thornton, N. J. Blair, J. A. Beckman, and S. J. Klein. 2017. *TRNSYS 18: A Transient System Simulation Program.* Solar Energy Laboratory, University of Wisconsin, Madison, USA. http://sel.me.wisc.edu/trnsys

[14] Michael Dahl Knudsen, Laurent Georges, Kristian Stenerud Skeie, and Steffen Petersen. 2021. Experimental test of a black-box economic model predictive control for residential space heating. *Applied Energy* 298 (2021), 117227. doi:10.1016/j.apenergy.2021.117227

[15] Felix Koch, Fabian Raisch, and Benjamin Tischler. 2026. Thermal-GEMs: Generalized Models for Building Thermal Dynamics. In *The 13th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation.* doi:10.1145/3744256.3812565

[16] Thomas Krug, Fabian Raisch, Dominik Aimer, Markus Wirnsberger, Ferdinand Sigg, Felix Koch, Benjamin Schäfer, and Benjamin Tischler. 2025. A Highly Configurable Framework for Large-Scale Thermal Building Data Generation to drive Machine Learning Research. *arXiv preprint arXiv:2512.00483* (2025).

[17] Thomas Krug, Fabian Raisch, Dominik Aimer, Markus Wirnsberger, Ferdinand Sigg, Benjamin Schäfer, and Benjamin Tischler. 2025. Builda: A thermal building data generation framework for transfer learning. In *2025 Annual Modeling and Simulation Conference (ANNSIM).* IEEE, 1–13.

[18] Tobias Loga, Nikolaus Diefenbach, and Rolf Born. 2011. *Deutsche Gebäudetypologie.* Wohnen und Umwelt, Darmstadt, Germany.

[19] 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).

[20] Henrik Madsen and JM Schultz. 1993. Short time determination of the heat dynamics of buildings. (1993).

[21] Ozan Baris Mulayim, Pengrui Quan, Liying Han, Xiaomin Ouyang, Dezhi Hong, Mario Bergés, and Mani Srivastava. 2024. Are Time Series Foundation Models Ready to Revolutionize Predictive Building Analytics?. In *Proceedings of the 11th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation.* 169–173.

[22] 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.

[23] Martin Pullinger, Jonathan Kilgour, Nigel Goddard, Niklas Berliner, Lynda Webb, Myroslava Dzikovska, Heather Lovell, Janek Mann, Charles Sutton, Janette Webb, et al. 2021. The IDEAL household energy dataset, electricity, gas, contextual sensor data and survey data for 255 UK homes. *Scientific Data* 8, 1 (2021), 146.

[24] Fabian Raisch, Timo Germann, J. Nathan Kutz, Christoph Goebel, and Benjamin Tischler. 2026. Transfer Learning for Neural Parameter Estimation applied to Building RC Models. arXiv:2604.05904 [eess.SY] https://arxiv.org/abs/2604.05904

[25] 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

[26] Fabian Raisch, Max Langtry, Felix Koch, Ruchi Choudhary, Christoph Goebel, and Benjamin Tischler. 2026. Adapting to change: A comparison of continual and transfer learning for modeling building thermal dynamics under concept drifts. *Energy and Buildings* 354 (2026), 116868. doi:10.1016/j.enbuild.2025.116868

[27] Igor Sartori, Harald Taxt Walnum, Kristian S. Skeie, Laurent Georges, Michael D. Knudsen, Peder Bacher, José Candanedo, Anna-Maria Sigounis, Anand Krishnan Prakash, Marco Pritoni, Jessica Granderson, Shiyu Yang, and Man Pun Wan. 2023. Sub-hourly measurement datasets from 6 real buildings: Energy use and indoor climate. *Data in Brief* (2023). doi:10.1016/j.dib.2023.109149

[28] Rawisha Serasinghe, Nicholas Long, and Jordan D. Clark. 2024. Parameter identification methods for low-order gray box building energy models: A critical review. *Energy and Buildings* 311 (2024), 114123. doi:10.1016/j.enbuild.2024.114123

[29] Dassault Systèmes. 2023. FMPy. https://github.com/CATIA-Systems/FMPy. Accessed: 2025-11-26.

[30] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).

[31] Michael Wetter, Wangda Zuo, Thierry S. Nouidui, and Xiufeng Pang. 2014. Modelica Buildings library. *Journal of Building Performance Simulation* 7, 4 (2014), 253–270. doi:10.1080/19401493.2013.765506

[32] Shiyu Yang, Man Pun Wan, Wanyu Chen, Bing Feng Ng, and Swapnil Dubey. 2021. Experiment study of machine-learning-based approximate model predictive control for energy-efficient building control. *Applied Energy* 288 (2021), 116648.

A Extended BuilDyn Description

BuilDyn is currently designed as a wrapper around BuilDa [16], also employing the BuilDa-specific FMU. In this part, we describe *BuilDyn* and BuilDa in more detail.

A.1 FMU Description

The BuilDa simulation framework [16, 17] enables the generation of high-fidelity operational data for residential buildings with systematically varied characteristics. The framework is implemented in Modelica and exported as a Functional Mock-up Unit (FMU), allowing users to configure building parameters directly in Python without modifying the underlying simulation model. The simulation model underlying BuilDa was validated in accordance with the ANSI/ASHRAE 140-2004 standard using the benchmark test cases TC600, TC900, TC600FF, and TC900FF. The model represents single-family houses with detailed envelope components, orientation-dependent windows, and a controllable heat/cool source sized to account for transmission and ventilation losses. Building properties such as insulation level, thermal mass, window-to-wall ratio, and floor area can be varied, and simulations can be performed for different weather locations. In addition, BuilDa provides a stochastic occupancy generator that produces user-specific temperature setpoints, internal heat gains, and window-opening schedules for natural ventilation based on the number of occupants and behavioral variability. These features enable the systematic generation of building datasets capturing variations in building characteristics, weather conditions, and occupant-driven operation.

A.2 Differences towards BuilDa

Table 1 contrasts the main differences between BuilDa and *BuilDyn*. As BuilDa is specifically designed for data generation, *BuilDyn* simplifies the simulation process and its integration into existing projects. Moreover, it introduces new functionalities, such as customizable excitations and variation distributions. For the scope of this paper, we utilize the FMU provided by BuilDa, which models a single-family home in central Europe. However, *BuilDyn* can be extended with additional FMUs modeling other building scenarios.

Table 1: Feature comparison BuilDa vs BuilDyn. Parentheses indicate that featurerequires substential effort implemented in the corresponding Tool/Extension.

	BuilDa	BuilDyn
Features		
Variations for Buildings	✓	✓
Parallel Simulation	✓	✓
Customizable Excitations	(✓)	✓
Controllers with FMU Feedback	✓	(✓)
Data Generation	✓	✓
Variation Distributions	×	✓
Usability		
Customizable Converter Functions	(✓)	✓
User-Friendly Python API	×	✓

B Extended BuilDyn Experiments

In this section, we present the use of *BuilDyn* in two different ways. First, we provide example code showing an application of the package and also illustrate the influence of variations on both the heating behavior and the energy consumption of a building.

B.1 Example simulation of building variations

We showcase *BuilDyn* using an example in which we simulate 40 PI-controlled buildings with identical construction under a Central European weather file, while varying the wall U-values. For each building, the indoor temperature and the heating load are recorded. Listing 1 illustrates how a single building instance is sampled and simulated using *BuilDyn*. For our example, we executed the last line of the code 40 times.

```
fmu = get_builda_fmu(use_controller=True)
observables = ["temperature", "load"]

buildyn = BuilDyn(fmu, observables)
wall_dist = GaussDistribution(mu=0.8, sigma=0.5)
buildyn.add_variation_distribution({"UExt",
    wall_dist})

# Get the simulation of a building that is
    randomly sampled from the wall_dist.
data = buildyn.sample_one(stop_time=15_552_000)
```

Listing 1: Simplified example code to generate one variated building simulation in BuilDyn.

The result is visualized in Figure 4, showing the monthly distributions of indoor temperature and heating load across buildings as box plots over six months. The plot illustrates that the variation of a single parameter, in this case the wall U-value, can strongly influence the thermal behavior of the buildings. This example demonstrates how easily *BuilDyn* can be configured to generate diverse residential building scenarios and produce substantial variability in thermal dynamics.

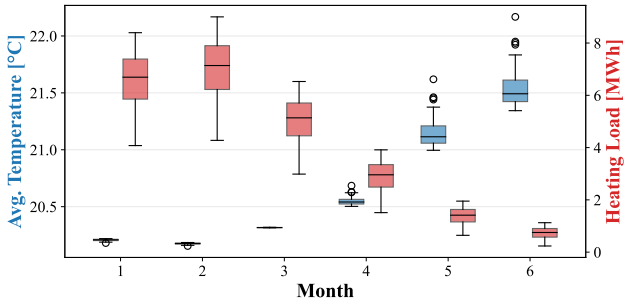


Figure 4: Heating load and indoor temperature distribution for 40 buildings.

B.2 Customizable variation distributions

In contrast to BuilDa, *BuilDyn* enables both grid-based parameter variation and sampling from user-defined parameter distribution. In this example, we sample 40 buildings from custom distributions

to show the difference. For that, we defined probability density functions (PDFs) over the sample space for the wall U-values and the floor area of a single-family home in Europe.

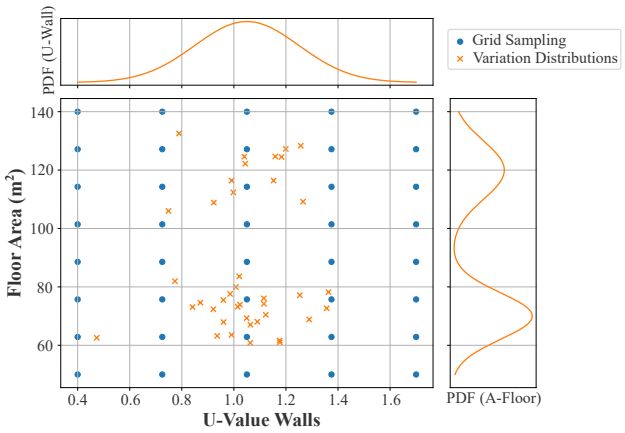


Figure 5: Sampling envelope parameter variations in *BuilDyn* from a custom distribution, an unimodal for the wall U-values, and a bimodal for the floor area.

Figure 5 visualizes the difference between the grid-like and the variation-distribution-based sampling. Notably, the variation distributions in *BuilDyn* are not restricted to Gaussian distributions, but can define all kinds of PDFs. Additionally, customizable variation distributions allow the reproduction of more realistic real-world building populations, which could support better benchmarking. These distributions are not restricted to continuous values over building envelope parameters, but also occupancy profiles or weather data files.

C Experiment Details

This section contains additional information to fully reproduce the experiments from Section 3. All excitations and data can be reproduced using the *BuilDyn* package.

C.1 ML Model Training

For the transformer models, we adopted the hyperparameters from [15]. The architecture consists of three transformer encoder blocks followed by two dense layers serving as the decoder. To account for the single-building modeling task, rather than training on data from multiple buildings, the hidden size of both the transformer and dense layers was reduced from 192 to 96. For a fair comparison with [15], we used a lookback of 96 (1 day) and a forecast horizon of 4; however, only the first forecasting step was considered in our experiments. Training was performed using the Adam optimizer [12]. Early stopping was applied based on performance on a validation set, defined as the final month of the first year.

C.2 Custom Excitation techniques

Section 3 showed three different excitation techniques to explore the building state-space better. This section showcases their implementation with pseudo-code. Notably, the respective distributions in the code are variable.

Poisson Excitation. The idea behind the Poisson excitation is to input the same heating signal for a varying period of time. The number of repetitions is determined by sampling from a Poisson distribution.

```
result = []
while len(result) < num_steps:
    v = random.uniform(0, 1)  # heating signal
    k = random.poisson(lam)
    result.extend([v] * k)
return result[:num_steps]
```

Listing 2: Pseudo-code for the poisson excitation signals.

Sinuoidal Excitation. The sinusoidal walker generates an excitation signal by chaining sinusoidal segments of randomly sampled frequency and amplitude, separated by brief steady-state holds. This should simulate different kinds of heating patterns.

```
result, phase = [], 0.0
mid = 0.5
while len(result) < num_steps:
    freq = sample(freq_dist)
    amp  = sample(amp_dist)
    hold = sample(steady_dist)
    omega = 2 * pi / freq
    for _ in range(freq):
        v = clip(mid + 0.5 * amp * sin(phase))
        result.append(v)
        phase += omega
    result.extend([result[-1]] * hold)
return result[:num_steps]
```

Listing 3: Pseudo-code for the sinusoidal excitation signals.

Ramp Excitation. This excitation strategy is repeatedly ramping linearly from a minimum to a maximum value and back, with brief steady-state holds at each peak and trough.

```
result = []
while len(result) < num_steps:
    freq = sample(freq_dist)
    step = 1 / (freq - 1)
    for i in range(freq):   # Ramp up
        result += [min + step * i]
    # hold high
    result.extend([1] * sample(steady_dist))
    for i in range(1, freq-1):  # Ramp down
        result.extend([1 - step * i])
    # hold low
    result.extend([0] * sample(steady_dist))
return result[:num_steps]
```

Listing 4: Pseudo-code for the ramp excitation signals.

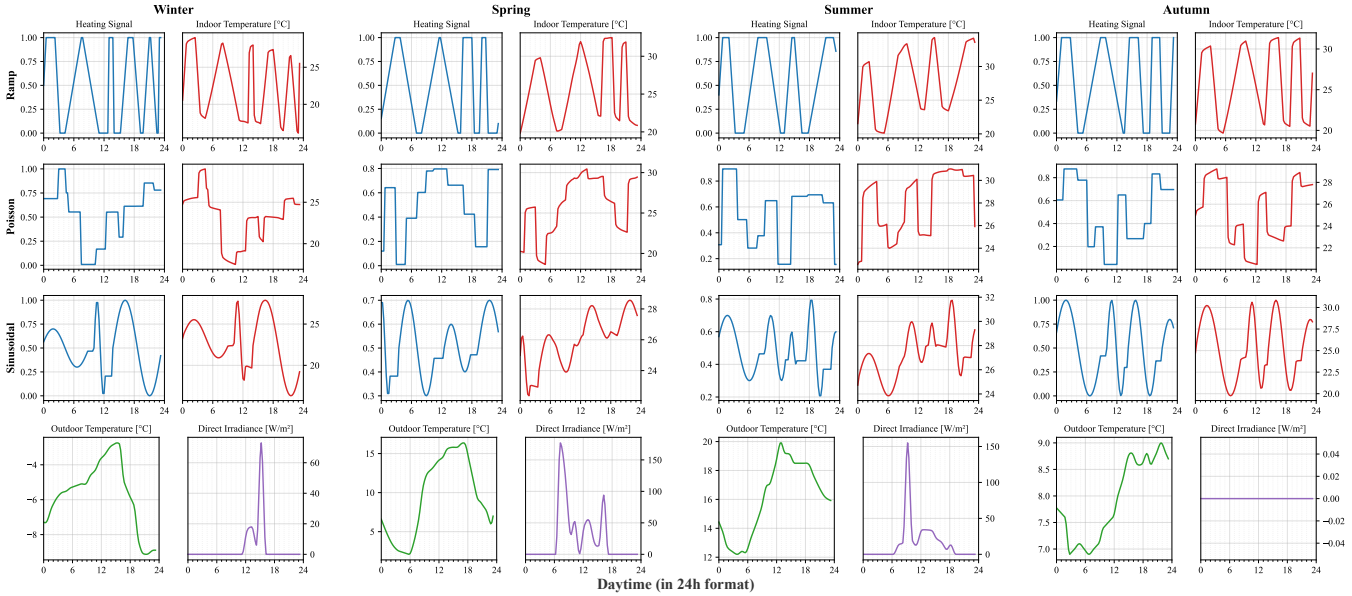


Figure 6: One day of excitation strategies that were used to create training data visualized for multiple seasons.

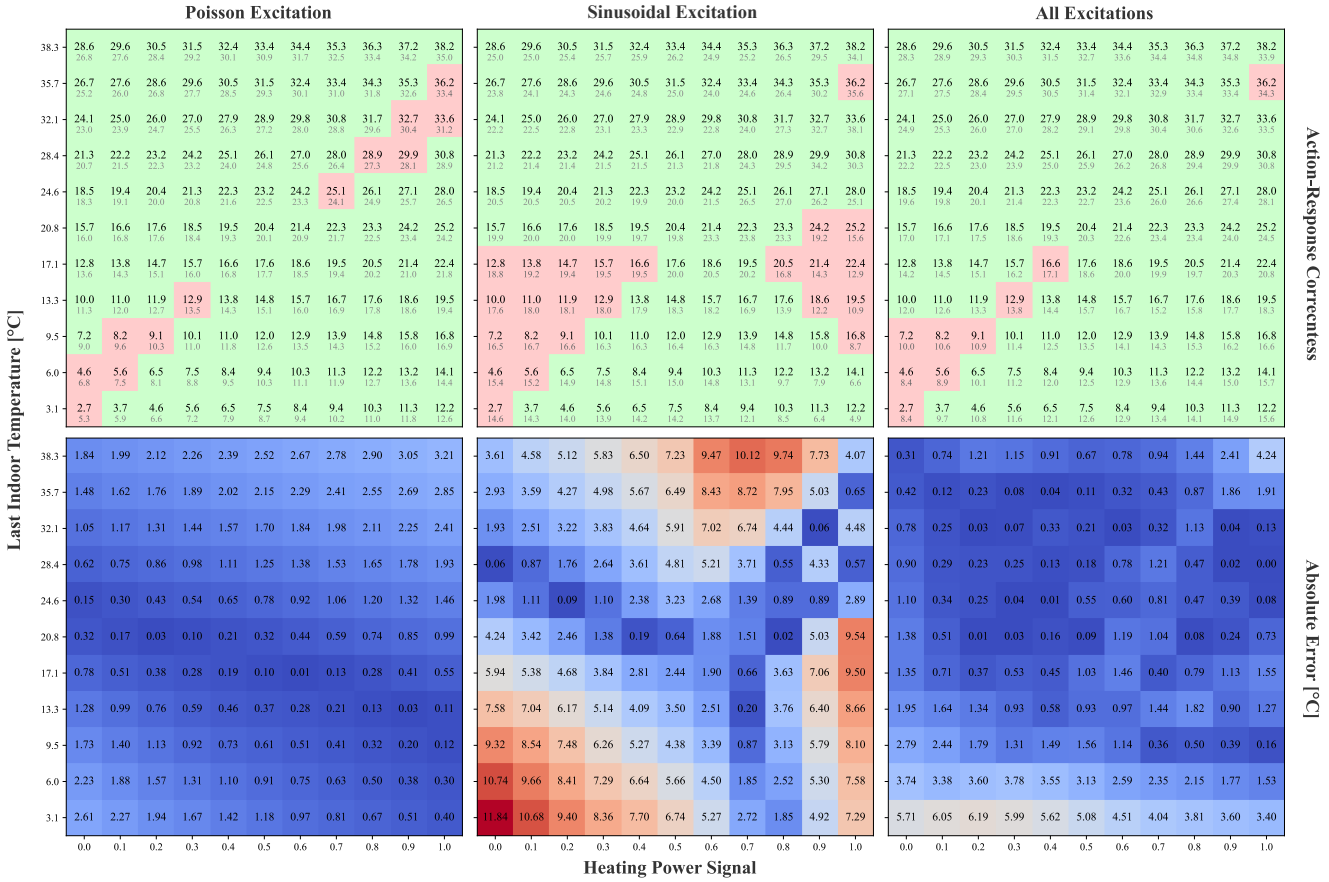


Figure 7: Complementary Absolute Error and Action-Response Correctness Heatmaps for other excitation techniques. The "All Excitations" strategy includes all three excitation techniques during training data generation.