NeuralBES: A Differentiable, Control-Aware Emulator for Scalable Building Energy Modeling

Ting-Yu Dai
Fujitsu Research of America
tdai@fujitsu.com

Takuya Kurihana
Fujitsu Research of America
tkurihana@fujitsu.com

Wing Yee Au
Fujitsu Research of America
WAu@fujitsu.com

HON YUNG WONG
Fujitsu Research of America
awong@fujitsu.com

Abstract

Demand-side flexibility, i.e., forecasting, shifting, and curtailing residential energy loads, depends on thermal models trusted across millions of heterogeneous buildings. Existing tools force a hard tradeoff: high-fidelity physics simulators such as EnergyPlus are accurate but sequential and require per-building calibration, while purely data-driven sequence models scale but abandon the physical structure that makes their predictions trustworthy.

We introduce NeuralBES (Building Energy Simulation), a differentiable emulator that resolves this tradeoff by parameterizing a resistance–capacitance (RC) based thermal model with a shared neural encoder: static building metadata such as floor area, vintage, and HVAC type is mapped to physically bounded capacitances, conductances, and equipment coefficients, which become the coefficients of a scalar linear recurrence solved via a log-space parallel scan, and a predictor–corrector loop closes the thermostat–temperature nonlinearity while preserving full-horizon gradient flow. Trained on the ResStock dataset across three climate zones, NeuralBES handles heterogeneous building archetypes, vintages, and climate zones within a single trained encoder, while black-box baselines produce statistically plausible but physically inconsistent trajectories. On the annual full-year rollout, NeuralBES reproduces realistic HVAC idle periods and duty cycles, which determine annual energy and peak-load estimates. It reduces the MAPE by more than half compared to an equally sized grey-box RC baseline that shares its metadata encoder, while an unconstrained LSTM attains lower pointwise error.

1 Introduction

Buildings account for roughly 40% of global energy consumption [9], and residential heating, ventilation, and air conditioning (HVAC), appliances, electric vehicles, and PV charging dominate this footprint; the surging demand of AI infrastructure is now adding a comparably fast-growing second load on the same grids [31]. Accurate, scalable models of demand-side behavior are therefore a prerequisite for grid decarbonization, and Building Energy Modeling (BEM) is the standard tool for producing them, combining building descriptors (geometry, construction, HVAC), operational signals (schedules, occupancy, setpoints), and outdoor weather to compute zone-level thermal loads for design, retrofit, and stock-level planning workflows [3, 29].

Classical BEM requires per-building calibration of dozens of parameters (wall layers, window properties, HVAC performance curves, infiltration coefficients), which is what gives it accuracy and what prevents it from scaling. Purely data-driven emulators scale but abandon the physical structure

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

that makes the outputs trustworthy under distribution shift (new climates, new control policies, new tariffs), which is precisely the regime where demand-flexibility studies need them most.

NeuralBES targets the middle. We address grid planners, utility analysts, MPC researchers, and policy modelers who need to simulate populations of buildings under counterfactual weather, tariffs, or control strategies. The user supplies only the metadata already available in tax-assessor records and utility rolls (floor area, vintage, HVAC type, ZIP code), and a shared encoder infers the parameters of an RC network, a circuit analogue in which capacitances C store heat and conductances UA (U-value $\times$ area A, the inverse of a thermal resistance R) set heat-loss rates (Appendix A), that would otherwise require a per-building calibration campaign. The result is a differentiable, physics-valid surrogate that trains once and predicts unseen buildings without per-building fitting or sacrificing the physical interpretability that makes predictions actionable.

Contributions. NeuralBES occupies a specific point in the BEM / ML design space, and we state it by contrasting with three adjacent families.

- **Compared to classical BEM (EnergyPlus [8], DOE-2 [39]):** we trade detailed envelope-level fidelity for two orders of magnitude fewer required inputs. The user supplies metadata already available in utility records rather than a full envelope specification, while we retain conservation-law structure and bounded, physically meaningful coefficients.
- **Compared to identified RC models [2, 4]:** we replace per-building least-squares identification with a single shared encoder that shares calibration across a population, enabling zero-shot prediction for buildings never seen during training.
- **Compared to generic state-space sequence models (S4 [17], Mamba [16], LRU [25]):** we constrain the state-space recurrence coefficients $\bar{A} = \exp(-P/C \cdot \Delta t)$ to be physically bounded functions of encoder-predicted capacitances and conductances rather than free learned parameters. We further wrap the resulting linear scan in a fixed-point loop that closes the thermostat–temperature nonlinearity, a mechanism outside the scope of linear SSMs.
- **Compared to black-box sequence models (LSTM [19], PatchTST [24], Roformer [33]):** attention-based sequence models incur $O(T^2)$ compute and memory in the rollout horizon, while our linear recurrence runs in $O(T)$ work and $O(\log T)$ depth via a parallel scan, which reduces training and inference time at long horizons. The resulting model remains physically valid (exclusive heating/cooling modes, bounded indoor temperature, correct sign of HVAC response) where purely data-driven baselines collapse to statistically plausible but physically inconsistent trajectories.

With these contributions and trade-offs, NeuralBES aims to provide functionality for several concrete use cases.

1. A utility planner who designs a demand-response program needs to know *when* heating and cooling operate across thousands of buildings for which only assessor records are available. A model that consistently predicts small spurious loads overstates peak demand and annual usage, so correctly predicting peak loads and duty cycles matters as much as low average error.
2. A retrofit or policy analyst who estimates envelope upgrades needs to evaluate counterfactuals, such as the effect of halving a building's heat-loss conductance UA; because NeuralBES exposes C and UA explicitly, such interventions can be specified directly and their effects checked for plausibility.
3. Researchers in model predictive control and tariff design need a fast, differentiable simulator that can evaluate many control scenarios.

In each case, NeuralBES is intended for population-level screening by emulating EnergyPlus under known weather and schedules.

2 Related Work

Identified RC thermal models. NeuralBES is most closely related to gray-box RC models, which represent building heat transfer with a small number of physically interpretable states and parameters. Foundational and follow-on studies showed that low-order thermal networks can capture building heat dynamics while remaining identifiable from data, and developed systematic procedures for selecting

Property	PINN [7]	Dissipative [12]	PCNN [10]	NeuralBES (ours)
Physics enforcement	soft penalty	stability regularizer	arch. constraints	analytical ODE
Parameters	per building	per building	per building	shared encoder
Evaluation	one building	one building	one building	held-out
Full-horizon gradients	via penalty	via BPTT	via BPTT	via parallel scan
Nonlinear controller	$\times$	$\times$	$\times$	predictor–corrector
Exposes C, UA at each t	$\times$	$\times$	partial	✓

Table 1: Conceptual positioning of NeuralBES relative to the closest physics-aware ML families. Rows name properties along the physics-enforcement, parameter-sharing, and gradient-flow axes; they are not accuracy numbers. C denotes thermal capacitance and UA thermal conductance (U-value $\times$ area A).

model order and fitting parameters for multi-zone and higher-fidelity networks [1, 2, 14, 4, 27, 26, 21, 20]. These methods are the source of the two-node abstraction we adopt, but are identified one building at a time: each calibration produces a parameter set valid only for that building, which is precisely the scaling bottleneck NeuralBES addresses by sharing identification across a population through a single encoder.

Physics-aware neural approaches. A growing body of work marries this physical backbone to modern machine learning. Neural ordinary differential equations parameterize continuous-time dynamics with neural networks and enable gradient-based calibration through an adjoint solver [6, 30], and have been applied directly to building thermodynamics for temperature and load forecasting [34]. A parallel line encodes thermodynamic structure into the architecture itself: physics-informed networks with RC-based losses [15, 7], physics-constrained models that guarantee dissipativity of the learned dynamics [12], and physically consistent neural networks that enforce monotonic HVAC response and energy conservation by construction [10, 11]. NeuralBES differs from these along three axes, summarized in Table 1: (i) we fix the two-node RC topology and learn only its coefficients, rather than learning the dynamics from scratch; (ii) the recurrence is solved analytically and in parallel rather than with a black-box ODE integrator or a soft physics penalty; and (iii) we wrap the linear solver in a predictor–corrector fixed-point loop that closes the thermostat–temperature saturation nonlinearity, a mechanism outside the scope of the linear and monotone architectures above. We do not attempt a head-to-head empirical comparison with these methods because their published implementations are fitted to one building or zone at a time and cannot condition on building metadata.

Parallel scans for linear recurrences. The computational machinery of NeuralBES is closest to recent work on parallel associative scans. Prefix-scan algorithms provide the general primitive for parallelizing cumulative associative operations in logarithmic depth [5]; Martin and Cundy first applied this primitive to linear recurrent neural networks [22], and structured state-space layers demonstrated that long linear dynamics can be trained efficiently with scan-based recurrences [16]. We adopt the log-space scan of Heinsen [18] because it stabilizes the long products of decay factors that arise when unrolling discretized RC dynamics; a sequential unroll would either require truncated backpropagation through time, which is known to bias long-horizon gradients [36], or store the entire 35,040-step trajectory for reverse mode, which is prohibitive at stock scale. Unlike generic state-space models, however, our recurrence coefficients are not free latent variables but building-specific capacitances, conductances, and infiltration terms produced by the encoder; NeuralBES uses the parallel-scan machinery developed for sequence modeling to solve a physically constrained, building-specific thermal ODE, joining two research threads that have evolved largely independently.

Training setting and population-scale surrogates. The training signal for NeuralBES is inherited from building simulation engines. DOE-2 and EnergyPlus established the modern zone/HVAC simulation workflow, including the coupling between thermal states and equipment operation that motivates our predictor–corrector update [39, 8]. At stock scale, ResStock provides the heterogeneous residential simulation corpus needed to learn shared structure across climates, vintages, and envelope characteristics [28, 37], and the End-Use Load Profiles project calibrated these simulations against millions of metered accounts, producing the benchmark dataset for population-scale residential energy modeling [38]. A parallel line of work builds pure-surrogate emulators over building populations without explicit physical structure [13, 35, 23]; NeuralBES shares the population-scale training

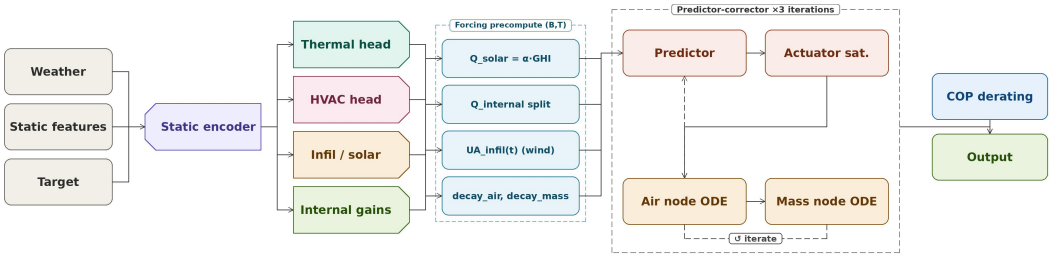


Figure 1: Architecture of NeuralBES. Static building features are encoded into a shared latent representation that conditions four parameter heads: the thermal head produces the RC capacitances C^{air}, C^{mass} and coefficients h^{am}, UA^{env}; the HVAC head produces heating and cooling capacities and COPs; the infiltration/solar head produces wind-dependent infiltration and solar gain coefficients; and the internal gains head maps occupancy schedules to $\dot{Q}^{\text{int}}$. Together with precomputed weather-driven forcings, setpoint schedules, and initial conditions, these parameters drive a predictor-corrector loop that solves the two-node RC thermal ODE via a parallel linear recurrence, iterated three times to resolve the nonlinear thermostat feedback. Electrical consumption is recovered by applying outdoor-temperature COP derating to the final HVAC thermal loads. Colors of equation variables throughout the paper correspond to the head that produces them.

objective with this surrogate literature but retains the conservation-law backbone, keeping the model identifiable, physically bounded, and usable downstream as a simulator or controller rather than as a fitted input–output mapping.

3 NeuralBES

3.1 Problem Formulation

We consider a batch of residential buildings $\mathcal{B} = \{1, \dots, B\}$ simulated over a sequence of 15-minute timesteps $\mathcal{T} = \{1, \dots, T\}$. Each building b is described by two kinds of inputs. The first is a static descriptor, which captures attributes that do not change over the simulation horizon: numerical covariates $\mathbf{s}_b^{\text{num}} \in \mathbb{R}^N$ (such as floor area, wall conductance, or window-to-wall ratio) and categorical covariates $\mathbf{s}_b^{\text{cat}} \in \mathbb{Z}^C$ (such as construction type or HVAC system category). The second is a time-varying exogenous signal $\mathbf{w}_{b,t} \in \mathbb{R}^D$, formed by concatenating weather variables, fractional occupancy and appliance schedules, and sinusoidal encodings of the time of day and year. At every timestep, the model produces three quantities, $\hat{\mathbf{y}}_{b,t} = \left[q_{b,t}^{\text{heat}},\ q_{b,t}^{\text{cool}},\ T_{b,t}^{\text{air}}\right] \in \mathbb{R}^3$: the heating and cooling energy densities q (in kWh/ft^2 per 15-minute interval, normalized by floor area) and the zone air temperature T^{air} (in °C). The heating and cooling setpoint schedules vary across buildings and over time and are supplied as inputs. Moreover, all exogenous inputs namely weather, schedules, and occupancy, are assumed to be known, following the conventional Building Energy Simulation setting. Predicting temperature jointly with the two energy channels is essential because in real buildings the HVAC load at any moment is determined by how far the indoor temperature has drifted from its setpoint; treating them as independent targets would discard this coupling.

The overall design of NeuralBES reflects this physical structure and is summarized in Figure 1. A neural encoder reads the static descriptor and produces both a latent building embedding $\mathbf{z}_b$ and a set of building-specific thermal parameters that populate a RC model. The time-varying signal $\mathbf{w}_{b,t}$ then acts as the external forcing on this model. Given these ingredients, inference proceeds in three stages: we (i) instantiate a two-node RC thermal model whose coefficients come from the encoder, (ii) solve the resulting linear recurrence in parallel across time using a log-space associative scan, and (iii) close the loop between indoor temperature and HVAC operation through a short predictor–corrector fixed-point iteration that enforces thermostat setpoints and equipment capacity limits. Appendix A glosses BEM and controls vocabulary (*setpoint*, *thermal mass*, *conductance*, COP, schedule) for readers unfamiliar with building simulation.

3.2 Neural Parameterization of the Two-Node RC Thermal Model

Each building is represented by a discretized two-node resistance–capacitance network with two capacitances and three conductances (a 3R2C model), assumed single-zone throughout. One node represents the zone air, whose temperature is what occupants and thermostats sense, while the other represents an aggregate building mass (walls, floors, furnishings) that stores and slowly releases heat. This two-node abstraction, standard in grey-box modeling, captures dominant short-term thermal behavior without a detailed envelope geometry. The network is formally identical to a two-capacitor electrical circuit: the thermal capacitances $C^{\text{air}}, C^{\text{mass}}$ play the role of capacitors, the conductances $h^{\text{am}}, UA^{\text{env}}, UA^{\text{infil}}$ play the role of inverse resistors, and the heat fluxes play the role of currents; readers familiar with linear circuit analysis will recognize the balances below as Kirchhoff's current law at each node. The continuous-time heat balance equates thermal capacitance times the rate of temperature change to the sum of incoming heat fluxes:

$$C_b^{\text{air}} \frac{dT_b^{\text{air}}}{dt} = \dot{Q}_b^{\text{sys}}(t) + \dot{Q}_b^{\text{int,conv}}(t) + UA_b^{\text{infil}}(t)\left(T_b^{\text{out}}(t) - T_b^{\text{air}}\right) + h_b^{\text{am}}\left(T_b^{\text{mass}} - T_b^{\text{air}}\right) \tag{1}$$

$$C_b^{\text{mass}} \frac{dT_b^{\text{mass}}}{dt} = \dot{Q}_b^{\text{sol}}(t) + \dot{Q}_b^{\text{int,rad}}(t) + h_b^{\text{am}}\left(T_b^{\text{air}} - T_b^{\text{mass}}\right) + UA_b^{\text{env}}\left(T_b^{\text{out}}(t) - T_b^{\text{mass}}\right) \tag{2}$$

Each right-hand side term corresponds to a distinct heat-transfer mechanism: HVAC delivery, infiltration with outdoor air, internal gains from occupancy and appliances, and solar gain on the mass. The shared air–mass coupling h^{am} is what allows the mass to buffer the air temperature and produce the characteristic thermal inertia of real buildings.

Parameter heads. Each head (color-coded in Figure 1) is a two-layer SiLU MLP conditioned on the static embedding $\mathbf{z}_b$, with a zero-initialized final layer and a bounded activation (softplus or sigmoid with head-specific scale) so that capacitances, conductances, capacities, and COPs remain strictly positive and in physically plausible ranges. The internal-gains head additionally consumes the per-timestep schedule vector. Full widths and activation scales are in Appendix C.

Exact discrete-time update. Over a simulation step of length $\Delta t = 900$ s we hold all forcing terms (HVAC load, internal gains, outdoor temperature, infiltration and solar inputs, and the other node's temperature) constant within the step. Under this assumption, the air-node balance in Eq. (1) reduces to a scalar linear ODE of the form

$$\frac{dT_b^{\text{air}}}{dt} = -\frac{P_{b,t}^{\text{air}}}{C_b^{\text{air}}}\left(T_b^{\text{air}} - T_{b,t}^{\text{air,eq}}\right), \qquad P_{b,t}^{\text{air}} = UA_b^{\text{infil}}(t) + h_b^{\text{am}}, \tag{3}$$

where $T_{b,t}^{\text{air,eq}}$ is the conductance-weighted sum of all forcing terms (Eq. (5) below). This ODE admits an exact closed-form solution over $[t, t+\Delta t]$, a geometric decay toward the instantaneous equilibrium:

$$T_{b,t}^{\text{air}} = \alpha_{b,t}^{\text{air}}\, T_{b,t-1}^{\text{air}} + \left(1 - \alpha_{b,t}^{\text{air}}\right) T_{b,t}^{\text{air,eq}}, \qquad \alpha_{b,t}^{\text{air}} = \exp\left(-\frac{P_{b,t}^{\text{air}}}{C_b^{\text{air}}}\,\Delta t\right). \tag{4}$$

Here $\alpha^{\text{air}} \in (0,1)$ is the closed-form solution of the scalar ODE with constant coefficients over one timestep, and P^{air} is the effective conductance sum that sets the thermal time constant. No numerical integrator (Euler, RK4) is needed.[1] The equilibrium temperature, toward which the air would relax if forcings were frozen, is the conductance-weighted average of all thermal inputs:

$$T_{b,t}^{\text{air,eq}} = \frac{\dot{Q}_b^{\text{sys}}(t) + \dot{Q}_b^{\text{int,conv}}(t) + UA_b^{\text{infil}}(t)\, T_{b,t}^{\text{out}} + h_b^{\text{am}}\, T_{b,t}^{\text{mass}}}{P_{b,t}^{\text{air}}}. \tag{5}$$

Introducing the affine term $\mathbf{b}_{b,t}^{\text{air}} = (1 - \alpha_{b,t}^{\text{air}})\, T_{b,t}^{\text{air,eq}}$ puts the update in the familiar state-space form $T_{b,t}^{\text{air}} = \alpha_{b,t}^{\text{air}}\, T_{b,t-1}^{\text{air}} + \mathbf{b}_{b,t}^{\text{air}}$. The mass node admits an identical derivation with its own $P_{b,t}^{\text{mass}} = h_b^{\text{am}} + UA_b^{\text{env}}$, capacitance C_b^{mass}, decay factor $\alpha_{b,t}^{\text{mass}} = \exp(-P_{b,t}^{\text{mass}}\Delta t / C_b^{\text{mass}})$, and equilibrium term. Both decay factors are strictly positive, bounded, and depend only on physical conductances and capacitances predicted by the encoder, making each recurrence a well-conditioned scalar linear system $x_t = \alpha_t x_{t-1} + b_t$. These two properties, physically exact (within the piecewise-constant-forcing assumption) and linear in the state, jointly enable the parallel scan of Section 3.3.

[1] In the implementation we clamp the exponent $P\Delta t/C$ at 4.6 (so $\alpha \geq 0.01$) as a float32 safety net against pathological encoder outputs during the first few training steps; this is a numerical guard, not a modeling choice, and is inactive once the encoder settles.

3.3 Parallel Linear Recurrence via Log-Space Scan

Once the RC coefficients are fixed, advancing the temperature trajectory amounts to evaluating a scalar linear recurrence of the form $x_t = \alpha_t x_{t-1} + b_t$ at each node. A naive implementation would roll this update out sequentially over T timesteps, preventing effective GPU parallelization and creating a severe bottleneck at long horizons. Since the recurrence is associative, we instead evaluate all timesteps in parallel via an associative-scan formulation, as is standard in structured state-space sequence models [5, 16]. Following Heinsen [18], the recurrence has the closed-form unrolling

$$T_{b,t} = \left(\prod_{s=1}^{t} \alpha_{b,s} \right) T_{b,0} + \sum_{s=1}^{t} \left(\prod_{r=s+1}^{t} \alpha_{b,r} \right) \mathbf{b}_{b,s}, \tag{6}$$

which expresses the state at time t as the attenuated initial condition plus a weighted sum of past forcings. Evaluating this directly is numerically fragile because cumulative products $\prod_s \alpha_s$ can underflow to machine zero at long horizons. The Heinsen log-space reformulation avoids this by working with cumulative log-decays:

$$\Lambda_{b,t} = \exp\left(\sum_{s=1}^{t} \ln \alpha_{b,s} \right), \qquad T_{b,t} = \Lambda_{b,t} \left(T_{b,0} + \sum_{s=1}^{t} \frac{\mathbf{b}_{b,s}}{\Lambda_{b,s}} \right). \tag{7}$$

Both the cumulative decay and the accumulated forcing are associative prefix operations and are computed with a standard $O(\log T)$-depth parallel scan. To prevent the rescaled forcings $e^{-L_{b,t}}$ from overflowing float32 when the cumulative decay becomes strongly negative over many timesteps, we execute the scan in adaptively-sized chunks and chain only the scalar boundary state between chunks; the chunk-size rule and its safety margin are in Appendix C.

3.4 Predictor–Corrector Thermostat Algorithm

The construction so far treats the HVAC heat delivery $\dot{Q}^{\text{sys}}$ as a known forcing, but in real operation this quantity is itself determined by the zone temperature: the thermostat activates heating when the air temperature would fall below the heating setpoint and cooling when it would rise above the cooling setpoint, and the delivered load is further limited by installed equipment capacity. This coupling between state and actuator is what makes classical simulators such as DOE-2 and EnergyPlus [39, 8] nonlinear in time, and it cannot be captured by a single linear scan. We therefore wrap the scan in a short fixed-point loop that alternates between (i) estimating the load the thermostat would demand at the current trajectory, (ii) clipping that demand by equipment capacity, and (iii) re-integrating the temperature trajectory under the clipped load. We use the names *predictor* and *corrector* for steps (i) and (iii) by analogy with the load–response logic inside EnergyPlus's zone predictor, not in the numerical-ODE sense; the ODE itself is already solved in closed form by Eq. (4), so what we iterate is the state–actuator coupling rather than an integrator. Three iterations are sufficient in practice ($N_{\text{iter}} = 3$).

Initial conditions. At $t = 0$ we teacher-force from ground truth during training and use a small MLP offset from the setpoint midpoint at inference; to suppress transients from a poorly chosen seed, the scan is run over $W = 16$ prepended warm-up steps (four hours at $\Delta t = 900\,\text{s}$) that replicate the first-step forcing, mirroring the steady-state relaxation performed implicitly by sequential simulators. Details are in Appendix C.

Fixed-point iteration. Each iteration consists of three steps that together close the state–actuator loop.

Step 1 (predictor). We ask a counterfactual: how much HVAC heat would be needed to keep the air exactly at setpoint, i.e., to set $dT^{\text{air}}/dt \equiv 0$. Setting the derivative to zero in the air balance and solving for the system load decomposes the answer into a passive disturbance term and a setpoint-dependent demand. The disturbance bundles all non-HVAC heat flows into the air node,

$$\dot{Q}^{\text{dist}}_{b,t} = \dot{Q}^{\text{int,conv}}_{b,t} + UA^{\text{infil}}_{b,t}\, T^{\text{out}}_{b,t} + h^{\text{am}}_{b}\, \hat{T}^{\text{mass}}_{b,t},$$

and the required heating and cooling loads are the additional heat needed to hold the air at the heating and cooling setpoints respectively:

$$\dot{Q}^{\text{req,heat}}_{b,t} = P^{\text{air}}_{b,t}\, T^{\text{heat,sp}}_{b,t} - \dot{Q}^{\text{dist}}_{b,t}, \qquad \dot{Q}^{\text{req,cool}}_{b,t} = P^{\text{air}}_{b,t}\, T^{\text{cool,sp}}_{b,t} - \dot{Q}^{\text{dist}}_{b,t}.$$

Step 2 (actuator saturation). The predicted demand is passed through a piecewise model of real thermostat and equipment behavior: if heating is required (positive demand) it is clipped to heating capacity; if cooling is required (negative demand) and no heating is requested it is clipped to cooling capacity; otherwise the HVAC idles:

$$\dot{Q}^{\text{sys}}_{b,t} = \begin{cases} \min\left(\dot{Q}^{\text{req,heat}}_{b,t},\ \dot{Q}^{\text{heat,cap}}_{b}\right) & \text{if } \dot{Q}^{\text{req,heat}}_{b,t} > 0, \\ \max\left(\dot{Q}^{\text{req,cool}}_{b,t},\ -\dot{Q}^{\text{cool,cap}}_{b}\right) & \text{if } \dot{Q}^{\text{req,cool}}_{b,t} < 0 \ \wedge\ \dot{Q}^{\text{req,heat}}_{b,t} \le 0, \\ 0 & \text{otherwise.} \end{cases}$$

This branch is the sole source of nonlinearity in the forward pass and produces realistic behaviors such as setpoint undershoot on very cold days when installed heating capacity is insufficient.

Step 3 (corrector). With $\dot{Q}^{\text{sys}}$ now fixed at its saturated value, the affine coefficients $\mathbf{b}^{\text{air}}, \mathbf{b}^{\text{mass}}$ are deterministic and both states are re-integrated with the parallel scan of Section 3.3:

$$\hat{T}^{\text{air}}_{b,\cdot} = \text{ParScan}\left(\alpha^{\text{air}}_{b,\cdot},\ \mathbf{b}^{\text{air}}_{b,\cdot}\left[\dot{Q}^{\text{sys}}\right],\ T^{\text{air}}_{b,0}\right), \qquad \hat{T}^{\text{mass}}_{b,\cdot} = \text{ParScan}\left(\alpha^{\text{mass}}_{b,\cdot},\ \mathbf{b}^{\text{mass}}_{b,\cdot}\left[\hat{T}^{\text{air}}\right],\ T^{\text{mass}}_{b,0}\right).$$

The updated trajectories feed back into Step 1 of the next iteration. Because the predictor and corrector operate on consistent thermal dynamics and only the saturation branch is nonlinear, the iteration contracts rapidly; three passes drive the residual below the noise floor of the underlying simulator.

4 Evaluation

4.1 Data and baselines

We train and evaluate on synthetic building-energy simulation data from ResStock [28], an NLR stock-level corpus that samples real U.S. housing characteristics and simulates each sampled dwelling with EnergyPlus. We distill subsets for California (CA), Texas (TX), and New York (NY) by stratified splitting over setpoint validity, plug-load usage, vintage, and building type, ensuring that every category appears in the training set to avoid cold-start embedding failures on held-out buildings. Each sample comes with static metadata, an hourly weather file, a 15-min schedule time series for equipment and occupant behavior, and a 15-min energy-usage ground-truth series over a full calendar year (sequence length $T = 4 \cdot 24 \cdot 365 = 35{,}040$). Training/validation/testing splits are 0.7/0.15/0.15 (2000/428/428 buildings per state), with a multi-state joint regime combining all three. All evaluations are on held-out buildings, and Appendix E additionally reports a leave-one-region-out experiment.

Evaluation protocol. We assess each model under two rollout horizons. All models are trained from scratch on 672-step windows that unfold each building's full year while no test-time fine-tuning is performed. The annual protocol is therefore an out-of-distribution-horizon evaluation roughly $52\times$ longer than the training context, while the weekly protocol is matched-horizon. The *annual* protocol ($35{,}040$ steps, one window per held-out building) is the primary evaluation, aligned with traditional simulation tools and with downstream applications such as annual energy-use intensity and retrofit ROI. The *weekly* protocol (672-step non-overlapping windows) is reported as a robustness check that exercises short-term thermal switching with frequent context resets. Per-region weekly numbers appear in Appendix E.

Baselines. We compare against six baselines spanning the inductive-bias spectrum (Table 5). All share the same input interface, i.e., weather, schedules, time encodings, and static features, and predict the same three output channels. All of them, including RC Grey-Box, process the static features with the same static-encoder architecture, and each model trains its own encoder weights end-to-end. The black-box models inject the resulting embedding through FiLM layers or embeddings, whereas RC Grey-Box and NeuralBES map it to physical coefficients. The exact widths and layer counts of each baseline model are shown in Appendix F.1.

RC Grey-Box baseline. Conventional grey-box RC models are calibrated separately for each building [2, 4], which is precisely the per-building fitting regime that NeuralBES is designed to avoid. To compare NeuralBES with an RC model that operates in the same population-level regime, we equip

Metric		DLinear	LSTM	PatchTST	Roformer	PI-Roformer	RC	**NeuralBES**
# Params (M)		1.3	0.98	3.0	20.5	11.0	0.96	**1.0**
MAPE (%)	Q_{heat}	$161.3_{\pm 39.8}$	$\mathbf{32.6_{\pm 2.4}}$	$52.4_{\pm 9.2}$	$67.8_{\pm 6.8}$	$65.9_{\pm 5.4}$	$86.5_{\pm 30.5}$	$41.5_{\pm 4.3}$
	Q_{cool}	$121.3_{\pm 22.9}$	$\mathbf{30.9_{\pm 4.9}}$	$46.2_{\pm 8.5}$	$85.3_{\pm 3.0}$	$80.6_{\pm 3.5}$	$92.5_{\pm 32.4}$	$41.3_{\pm 0.4}$
RMSE (°C)	T_{in}	$3.43_{\pm 0.83}$	$\mathbf{1.30_{\pm 0.10}}$	$1.68_{\pm 0.13}$	$3.26_{\pm 0.08}$	$2.92_{\pm 0.41}$	$4.05_{\pm 0.52}$	$2.79_{\pm 0.03}$
Mode excl. [%]		$26.1_{\pm 27.2}$	$32.4_{\pm 9.6}$	$45.5_{\pm 22.2}$	$24.4_{\pm 8.5}$	$31.0_{\pm 28.1}$	$\mathbf{100.0_{\pm 0.0}}$	$\mathbf{100.0_{\pm 0.0}}$
Bounds [%]		$73.8_{\pm 27.2}$	$67.2_{\pm 9.4}$	$54.3_{\pm 22.2}$	$75.6_{\pm 8.5}$	$68.8_{\pm 27.9}$	$98.6_{\pm 1.4}$	$\mathbf{99.7_{\pm 0.1}}$

Table 2: Annual full-year (35,040-step) rollout on the multi-region ResStock test set. NeuralBES and RC are the only models with structural physics validity (100% mode exclusivity, 99.7% bounds adherence). The LSTM shown here achieves the lowest raw error but produces physically inadmissible trajectories (mode exclusivity 32.4%, bounds adherence 67.2%). Among physics-valid models (NeuralBES vs. RC) at parameter parity, NeuralBES more than halves RC's MAPE. Weekly-rollout numbers, results for a larger LSTM, and per-region breakdowns are shown in Appendix E.

RC Grey-Box with the shared static encoder, so that both models infer building-specific coefficients of the same 3R2C network from metadata alone. The encoder of RC Grey-Box predicts the coefficients that per-building studies typically identify (C^{air}, C^{mass}, h^{am}, UA^{env}, $\dot{Q}^{\text{heat,cap}}$, $\dot{Q}^{\text{cool,cap}}$, solar aperture, base ACH). Internal gains are modeled by a single population-level linear map of the 17 schedule channels, the COPs and setpoint band are fixed, and the temperature is rolled out sequentially. The two models therefore share their inputs, network topology, and parameter budget, and differ only in how the RC dynamics are simulated: NeuralBES adds the predictor–corrector thermostat, schedule-aware setpoints, building-conditioned internal gains, wind-driven infiltration, COP derating, and the parallel log-space scan. The comparison between RC Grey-Box and NeuralBES thus measures the benefit of these components, with the predictor–corrector thermostat at their core, over a population-level RC model with identical inputs; it does not separate their individual contributions.

4.2 Results and Analysis

Overall accuracy. We evaluate the overall accuracy separately for the three output channels, as shown in Table 2. As a purely data-driven model that is not constrained by physics, the LSTM achieves the best pointwise accuracy on all three channels, with 9–10 pp lower HVAC MAPE and $1.49\,^{\circ}$C lower T_{in} RMSE than NeuralBES. A black-box model of this kind is therefore preferable when only pointwise predictions are required. Roformer (20.5 M) is worse than NeuralBES on every dimension. At parameter parity, NeuralBES more than halves the MAPE of the grey-box RC and cuts T_{in} RMSE by $1.3\,^{\circ}$C.

Physics validity. The bottom block of Table 2 reports two thermodynamic diagnostics: *mode exclusivity* (fraction of timesteps with no simultaneous heating and cooling, declared simultaneous when both $\hat{Q}_{\text{heat}}$ and $\hat{Q}_{\text{cool}}$ exceed $\epsilon = 10^{-3}$ in normalized load units) and *bounds adherence* (fraction of T_{in} within the physically plausible envelope). NeuralBES achieves 100% mode exclusivity and $99.7 \pm 0.1\%$ bounds adherence; RC matches it at 100% / 98.6% but pays $86.5/92.5\%$ MAPE for the rigidity of fixed COPs, a fixed setpoint band, and no schedule-aware dynamics. The data-driven baselines reach only 24.4–45.5% mode exclusivity and 54.3–75.6% bounds adherence, with large variation across seeds: without physical structure, nothing prevents them from predicting heating and cooling at the same time or indoor temperatures outside the feasible range.

Post-processing the black-box outputs. Nonetheless, these violations by black-box models can be removed. We experimented with post-processing the LSTM output, either by predicting a single signed load or by clipping it to the feasible range. A single signed load rules out simultaneous heating and cooling, and clipping keeps the output within the feasible range, so the two corrections together enforce both diagnostics by construction. Such post-processing, however, only compensates for violations after they occur rather than preventing them. Because the constraints are imposed on the outputs rather than learned, the clipped loads still do not agree with the temperature the same network predicts, and each new constraint needs its own hand-designed correction. Since the correction also acts on each timestep independently, it cannot repair the temporal pattern of the loads. When the true heating load is zero, NeuralBES predicts zero heating in 94.8% of timesteps, compared with 92.1% for the post-processed LSTM. Because clipping makes every residual leak positive, these errors accumulate rather than cancel, amounting to 4.6% of the true annual heating energy for the LSTM,

versus 3.7% for NeuralBES. NeuralBES also matches the heating off-time more closely (76.0% for NeuralBES versus 75.1% in the ground truth), whereas the LSTM underestimates it.

Sensitivity analysis. Figure 2 stress-tests envelope behavior by uniformly perturbing T_{out} by ± 2 and ± 5 °C and measuring the mean shift in predicted T_{in}. NeuralBES and RC give the physically expected monotone response; Roformer and PI-Roformer over-react with implausibly leaky envelopes; PatchTST returns a flat zero response, i.e. its T_{in} prediction is *formally independent* of outdoor temperature. A leakier-than-physical envelope is bounded over a week but accumulates a multi-degree drift over a year (Appendix E). We hypothesize that the downstream tasks motivating this work (MPC, tariff design under demand response, heatwave resilience, and stock-level retrofit ROI) place as much weight on the timing of equipment operation as on pointwise MAPE, because mode-violating loads pollute annual cost integrals and small leakiness errors compound across the year. Quantifying this directly, for instance via annual electricity-cost MAE under a time-of-use tariff applied to predicted load shapes, is left to future work.

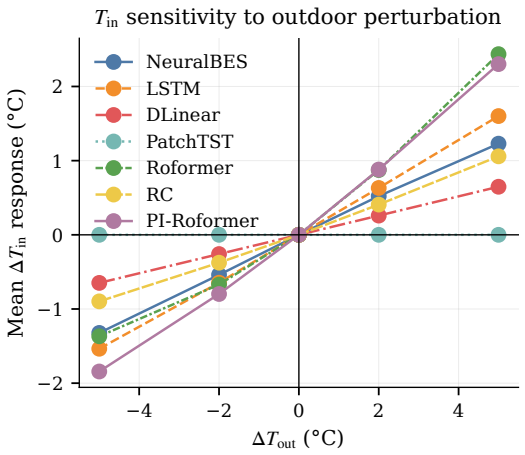


Figure 2: T_{in} sensitivity to a uniform T_{out} perturbation of $\{-5, -2, 0, +2, +5\}$ °C. NeuralBES and RC give the physically expected monotone slope; Roformer and PI-Roformer over-react (indoor temperature shifts by more than 2 °C for a 5 °C outdoor perturbation, implying an implausibly leaky envelope); DLinear's slope is shallow but in the right direction; PatchTST returns a flat zero response at every perturbation, i.e. its T_{in} prediction is formally independent of T_{out}.

Qualitative trajectory comparison. Figure 3 makes the qualitative content of the physics-validity numbers visible. NeuralBES emits identically zero on load channels in seasons with no heating or cooling and produces discrete spikes at thermostat events; baselines emit either constant non-zero offsets (DLinear, RC) or low-amplitude noise (Roformer winter Q_{heat}; PatchTST and Roformer summer Q_{heat}). On T_{in}, NeuralBES tracks envelope-driven excursions in summer and fall while LSTM and PatchTST flatten toward the setpoint mean. These predictions are statistically defensible but physically nonsensical, exactly what the mode-exclusivity and bounds-adherence diagnostics measure. The case study in Appendix E shows the most extreme instance: Roformer flatlines at ~ 21 °C while the ground truth swings to ~ 26 °C in fall.

Hybrid comparison. The Roformer / PI-Roformer pair isolates the effect of attaching a physics head to a data-driven trunk. PI-Roformer narrows the HVAC MAPE gap only marginally and gives a small T_{in} RMSE improvement, while mode exclusivity and bounds adherence remain within seed-to-seed spread of Roformer and far below NeuralBES. Notably, PI-Roformer's mode-exclusivity ranking inverts between horizons: it scores 31.0% on the annual rollout (better than Roformer's 24.4%) but only 2.75% on the weekly Multi rollout (far below Roformer's 43.38%, Appendix E), consistent with a physics head that reduces residual energy-conservation error over long integration windows but provides no benefit, and possibly hurts, on short-window switching. A soft heat-balance residual on a black-box trunk does not recover the switching dynamics: it regularizes the trunk but does not explicitly simulate the thermostat, whereas the parallel log-space scan in NeuralBES carries gradients across the full training window and the predictor–corrector closes the thermostat-to-temperature loop on a dissipative state.

5 Conclusion

We introduce NeuralBES, a differentiable building-energy emulator that parameterizes a two-node RC thermal model with a shared encoder and solves the resulting linear recurrence in parallel via a log-space associative scan, wrapped in a predictor–corrector loop that closes the thermostat–temperature nonlinearity. On the multi-region ResStock annual-rollout benchmark, NeuralBES is less accurate

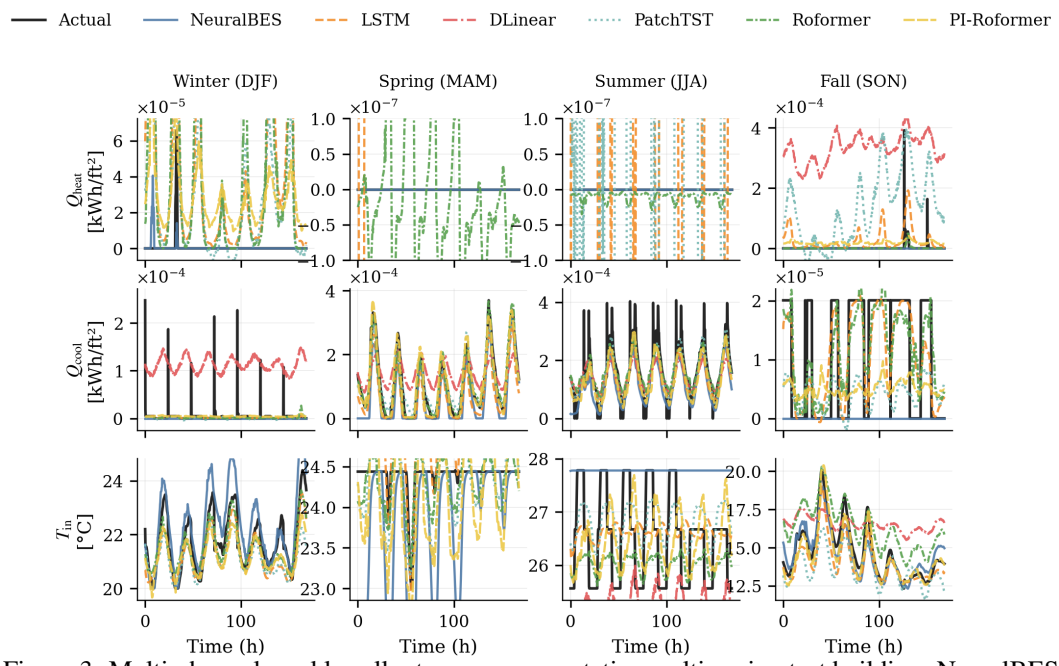


Figure 3: Multi-channel weekly rollouts on a representative multi-region test building. NeuralBES (blue) produces sparse, on/off load trajectories aligned with thermostat events. Baselines emit visibly noisy or near-constant predictions that violate the actual loads' seasonal sparsity: DLinear (red) holds a persistent winter Q_{cool}; Roformer (green) shows high-frequency winter Q_{heat} oscillations orders of magnitude below the real events (note the 10^{-7} vs. 10^{-4} axis scales); PatchTST emits non-zero summer Q_{heat}. On T_{in}, NeuralBES tracks envelope-driven excursions while LSTM and PatchTST flatten near the setpoint mean.

than black-box models in pointwise errors, but it more faithfully reproduces the operating state of the HVAC equipment, namely when it is idle and its duty cycle, which are the quantities that load-shape, tariff, and capacity studies depend on. It also substantially outperforms the comparably structured grey-box RC at parameter parity ($\sim 41\%$ vs. $\sim 89\%$ MAPE on the HVAC channels). First-principles thermodynamics and scalable neural sequence modeling share the same forward pass.

Limitations and future work. NeuralBES is a surrogate for a physics-based simulator and has not itself been validated against metered data or with forecast inputs. It inherits the limitations of the ResStock EnergyPlus simulations it emulates while removing the need for their detailed inputs. It also assumes known weather, schedules, and occupancy. Its scalability advantage lies in its inputs; throughput on large stocks is not benchmarked. The learned C and UA are effective parameters, as ResStock provides no ground-truth RC values. The thermostat model represents single-stage equipment only, and results are stratified by region but not by building type.

Two further limitations frame the next steps. First, NeuralBES resolves only the sensible thermal dynamics of the zone air and building mass and does not explicitly model the domestic hot-water system, which is absent from the heat-balance equation that drives the analytical solver and must currently be absorbed through the generic residual channel; a dedicated water-side sub-model inside the same differentiable integrator is a natural extension. Second, scalability along the *physics* axis is bounded by the equations we commit to: adding phenomena such as multi-zone coupling, refrigerant cycles, or stratified storage requires editing the governing ODE rather than widening the network. The more interesting research target going forward is how to extend the *data-driven* component so that it can absorb phenomena hard to express analytically while keeping the interpretability and generalization properties that the physical backbone provides.

References

[1] Klaus Kaae Andersen, Henrik Madsen, and Lars Henrik Hansen. Modelling the heat dynamics of a building using stochastic differential equations. *Energy and Buildings*, 31(1):13–24, 2000. doi: 10.1016/S0378-7788(98)00069-3.

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

[3] Brett Bass, Joshua New, and William Copeland. Potential energy, demand, emissions, and cost savings distributions for buildings in a utility's service area. *Energies*, 14(1):132, 2020.

[4] Thomas Berthou, Pascal Stabat, Raphaël Salvazet, and Dominique Marchio. Development and validation of a gray box model to predict thermal behavior of occupied office buildings. *Energy and Buildings*, 74:91–100, 2014. doi: 10.1016/j.enbuild.2014.01.038.

[5] Guy E Blelloch. Prefix sums and their applications. 1990.

[6] Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. *Advances in neural information processing systems*, 31, 2018.

[7] Yongbao Chen, Qiguo Yang, Zhe Chen, Chengchu Yan, Shu Zeng, and Mingkun Dai. Physics-informed neural networks for building thermal modeling and demand response control. *Building and Environment*, 234:110149, 2023.

[8] 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. EnergyPlus: creating a new-generation building energy simulation program. *Energy and buildings*, 33(4):319–331, 2001.

[9] Pierre Dechamps. The IEA World Energy Outlook 2022–a brief analysis and implications. *European Energy & Climate Journal*, 11(3):100–103, 2023.

[10] Loris Di Natale, Bratislav Svetozarevic, Philipp Heer, and Colin N Jones. Physically consistent neural networks for building thermal modeling: Theory and analysis. *Applied Energy*, 325: 119806, 2022. doi: 10.1016/j.apenergy.2022.119806.

[11] Loris Di Natale, Bratislav Svetozarevic, Philipp Heer, and Colin Neil Jones. Towards scalable physically consistent neural networks: An application to data-driven multi-zone thermal building models. *Applied Energy*, 340:121071, 2023.

[12] Ján Drgoňa, Aaron R Tuor, Vikas Chandan, and Draguna L Vrabie. Physics-constrained deep learning of multi-zone building thermal dynamics. *Energy and Buildings*, 243:110992, 2021.

[13] Richard E Edwards, Joshua New, Lynne E Parker, Borui Cui, and Jin Dong. Constructing large scale surrogate models from big data and artificial intelligence. *Applied energy*, 202:685–699, 2017.

[14] G Fraisse, C Viardot, O Lafabrie, and G Achard. Development of a simplified and accurate building model based on electrical analogy. *Energy and Buildings*, 34(10):1017–1031, 2002. doi: 10.1016/S0378-7788(02)00019-1.

[15] Gargya Gokhale, Bert Claessens, and Chris Develder. Physics informed neural networks for control oriented thermal modeling of buildings. *Applied Energy*, 314:118852, 2022.

[16] Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*, 2023.

[17] Albert Gu, Karan Goel, and Christopher Ré. Efficiently modeling long sequences with structured state spaces. In *International Conference on Learning Representations (ICLR)*, 2022.

[18] Franz A Heinsen. Efficient parallelization of a ubiquitous sequential computation. *arXiv preprint arXiv:2311.06281*, 2023.

[19] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8): 1735–1780, 1997.

[20] Martin Heine Kristensen, Rasmus Elbæk Hedegaard, and Steffen Petersen. Hierarchical calibration of archetypes for urban building energy modeling. *Energy and Buildings*, 175: 219–234, 2018.

[21] Yanfei Li, Zheng O'Neill, Liang Zhang, Jianli Chen, Piljae Im, and Jason DeGraw. Grey-box modeling and application for building energy simulations-a critical review. *Renewable and Sustainable Energy Reviews*, 146:111174, 2021.

[22] Eric Martin and Chris Cundy. Parallelizing linear recurrent neural nets over sequence length. In *International Conference on Learning Representations (ICLR)*, 2018.

[23] Clayton Miller. More buildings make more generalizable models—benchmarking prediction methods on open electrical meter data. *Machine Learning and Knowledge Extraction*, 1(3): 974–993, 2019.

[24] Yuqi Nie, Nam H Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. *arXiv preprint arXiv:2211.14730*, 2022.

[25] Antonio Orvieto, Samuel L Smith, Albert Gu, Anushan Fernando, Caglar Gulcehre, Razvan Pascanu, and Soham De. Resurrecting recurrent neural networks for long sequences. In *International conference on machine learning*, pages 26670–26698. PMLR, 2023.

[26] Loïc Raillon and Christian Ghiaus. Study of error propagation in the transformations of dynamic thermal models of buildings. *Journal of Control Science and Engineering*, 2017:5636145, 2017. doi: 10.1155/2017/5636145.

[27] Alfonso P Ramallo-González, Matthew E Eames, and David A Coley. Lumped parameter models for building thermal modelling: An analytic approach to simplifying complex multi-layered constructions. *Energy and Buildings*, 60:174–184, 2013. doi: 10.1016/j.enbuild.2013.01.014.

[28] Janet Reyna, Anthony Fontanini, Elaina Present, Lixi Liu, Rajendra Adhikari, Carlo Bianchi, Jes Brossman, Rohit Chintala, Kenya Clark, Chioke Harris, et al. Resstock technical reference documentation (v. 3.3. 0). Technical report, National Renewable Energy Laboratory (NREL), Golden, CO (United States), 2025.

[29] Amir Roth. Building energy modeling 101: Stock-level analysis use case. U.S. Department of Energy, Office of Energy Efficiency and Renewable Energy, Building Technologies Office, April 2017. URL `https://www.energy.gov/eere/buildings/articles/building-energy-modeling-101-stock-level-analysis-use-case`.

[30] Yulia Rubanova, Ricky TQ Chen, and David K Duvenaud. Latent ordinary differential equations for irregularly-sampled time series. *Advances in neural information processing systems*, 32, 2019.

[31] Arman Shehabi, Alex Newkirk, Sarah J Smith, Alex Hubbard, Nuoa Lei, Md Abu Bakar Siddik, Billie Holecek, Jonathan Koomey, Eric Masanet, and Dale Sartor. 2024 United States Data Center Energy Usage Report. 2024.

[32] Jimmy TH Smith, Andrew Warrington, and Scott W Linderman. Simplified state space layers for sequence modeling. *arXiv preprint arXiv:2208.04933*, 2022.

[33] Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

[34] Vincent Taboga, Clement Gehring, Mathieu Le Cam, Hanane Dagdougui, and Pierre-Luc Bacon. Neural differential equations for temperature control in buildings under demand response programs. *Applied Energy*, 368:123433, 2024.

[35] Paul Westermann, Matthias Welzel, and Ralph Evins. Using a deep temporal convolutional network as a building energy surrogate model that spans multiple climate zones. *Applied Energy*, 278:115563, 2020.

[36] Ronald J Williams and Jing Peng. An efficient gradient-based algorithm for on-line training of recurrent network trajectories. *Neural Computation*, 2(4):490–501, 1990. doi: 10.1162/neco.1990.2.4.490.

[37] Eric Wilson, Craig Christensen, Scott Horowitz, Joseph Robertson, and Jeff Maguire. Energy efficiency potential in the U.S. single-family housing stock. Technical Report NREL/TP-5500-68670, National Renewable Energy Laboratory (NREL), Golden, CO (United States), 2017.

[38] Eric JH Wilson, Andrew Parker, Anthony Fontanini, Elaina Present, Janet L Reyna, Rajendra Adhikari, Carlo Bianchi, Christopher CaraDonna, Matthew Dahlhausen, Janghyun Kim, et al. End-use load profiles for the U.S. building stock: Methodology and results of model calibration, validation, and uncertainty quantification. Technical Report NREL/TP-5500-80889, National Renewable Energy Laboratory (NREL), Golden, CO (United States), 2022.

[39] FC Winkelmann, BE Birdsall, WF Buhl, KL Ellington, AE Erdem, JJ Hirsch, and Stephen Gates. Doe-2 supplement: version 2.1 e. Technical report, Lawrence Berkeley Lab., CA (United States); Hirsch (James J.) and Associates . . . , 1993.

[40] Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting? In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pages 11121–11128, 2023.

A BEM Primer for ML Readers

This appendix is written for readers who know ML but have never opened EnergyPlus. Its purpose is to make the rest of the paper legible by making the operational vocabulary of building energy modeling concrete.

What a zone is. A *thermal zone* is a single conditioned-air volume that a building simulator tracks as one state. A whole detached house is typically one zone; a commercial floor with multiple thermostats may be many. In this paper every dwelling is a single zone, following the default ResStock configuration for residential single-family homes. A zone has exactly one air temperature, one thermostat, and one setpoint schedule at any instant, and receives heat from four categorical sources: the envelope (walls / windows / roof), infiltration (air leakage), internal gains (people, lights, plug loads), and the HVAC system.

What the envelope does. The *envelope* is the set of opaque and glazed surfaces separating the zone from outdoors. Its thermal behavior is characterized by conductance (how much heat flows per degree of temperature difference) and thermal mass (how much heat it can store before its own surface temperature responds). In a high-fidelity simulator, the envelope is resolved layer by layer (gypsum, insulation, framing, sheathing, cladding), and each wall's conduction transfer function is computed from manufacturer-level material properties. A two-node RC network lumps all of this into a single C^{mass} and a single UA^{env}, which is both the source of its tractability and the source of its error relative to a high-fidelity simulator.

What a schedule is. Because residential energy use is dominated by occupant behavior, a simulator needs a proxy for that behavior. A *schedule* is a 15-minute or hourly fractional time series (between 0 and 1) that represents occupancy, appliance use, lighting, or HVAC-mode selection. ResStock draws schedules from a stochastic model calibrated to time-use surveys and appliance metering studies, so each building in the dataset has its own schedules but they are statistically representative of U.S. residences. NeuralBES treats schedules as exogenous covariates, exactly as EnergyPlus does: they drive internal gains and setpoint trajectories but are not model outputs.

Setpoint and setback. A residential thermostat tracks two scheduled temperatures: a heating setpoint $T^{\text{heat,sp}}$ (typically 20–22 °C) and a cooling setpoint $T^{\text{cool,sp}}$ (typically 24–26 °C). Heating activates when indoor temperature would drop below $T^{\text{heat,sp}}$, cooling activates when it would rise above $T^{\text{cool,sp}}$, and between the two the HVAC system idles and the indoor temperature floats. A *setback* or *setup* is a scheduled relaxation of the setpoint (e.g. heating setpoint dropped to 18 °C at

night), and is what programmable thermostats implement. Most residential equipment runs single-stage (on/off at full capacity); the actuator-saturation step of Section 3.4 imposes the analogous capacity limit on the predicted load.

COP, sensible load, and latent load. HVAC equipment delivers heat to or removes heat from the zone air; the amount of electrical energy required to do so is the load divided by the *coefficient of performance* (COP). For a heat pump, COP is roughly 2–4 and degrades sharply at extreme outdoor temperatures ("derating"); for a resistive heater it is exactly 1. We model only the *sensible* component of HVAC heat transfer, the part that changes air temperature. The *latent* component (dehumidification) is present in real systems but not in our two-node abstraction, which is why our predictions of cooling energy are expected to under-state total cooling electricity in humid climates; this is a limitation shared with the entire 3R2C literature.

What ResStock does and does not simulate. ResStock is a stock-level sampler that runs EnergyPlus on 550k representative U.S. single-family homes with stochastic occupant schedules, actual TMY weather, and realistic distributions over construction vintage, HVAC type, and equipment age. It simulates: sensible and latent zone loads, HVAC runtime and electricity at 15-minute resolution, water-heater cycles, and plug-load timeseries. It does not simulate: real occupant decisions (overrides of programmable schedules), real equipment faults, ductwork commissioning, or short-cycling below 15-minute resolution. The End-Use Load Profiles project then calibrates aggregate ResStock outputs against metered consumption at the utility level, which is what makes the dataset usable as ground truth for population-scale emulators. NeuralBES emulates ResStock, not metered reality; the scope of the claim in this paper is correspondingly the scope of the ResStock simulator.

Why 15-minute resolution matters. Residential thermostats cycle on timescales of 5–30 minutes. At hourly resolution, the cycling is invisible (averaged away), and load forecasts look smooth but miss the peaks that matter for grid congestion and capacity planning. At 1-minute resolution, one captures individual compressor starts but the additional fidelity is not recoverable from hourly weather files. 15 minutes is the granularity at which grid operators, demand-response programs, and ASHRAE schedules are typically defined, and is the native resolution of ResStock timeseries outputs.

B Heat Balance and SSM Correspondence

This appendix grounds NeuralBES in two complementary formalisms. Section B.1 derives the two-node balance from the full EnergyPlus zone-air heat balance, and Section B.2 maps the resulting recurrence onto the standard structured state-space model (SSM) notation used by S4, S5, and Mamba.

B.1 Heat Balance Derivation

The two-node balance of Section 3.2 is a reduction of the zone air heat balance equation as solved by EnergyPlus. The full equation is:

$$C_z \frac{dT_z}{dt} = \sum_{i=1}^{N_{sl}} \dot{Q}_i + \sum_{i=1}^{N_{surfaces}} h_i A_i (T_{si} - T_z) + \sum_{i=1}^{N_{zones}} \dot{m}_i C_p (T_{zi} - T_z) + \dot{m}_{inf} C_p (T_\infty - T_z) + \dot{Q}_{sys} \tag{8}$$

where

- $\sum_{i=1}^{N_{sl}} \dot{Q}_i$ = sum of the convective internal loads.
- $\sum_{i=1}^{N_{surfaces}} h_i A_i (T_{si} - T_z)$ = convective heat transfer from the zone surfaces.
- $\dot{m}_{inf} C_p (T_\infty - T_z)$ = heat transfer due to infiltration of outside air.
- $\sum_{i=1}^{N_{zones}} \dot{m}_i C_p (T_{zi} - T_z)$ = heat transfer due to interzone air mixing.
- $\dot{Q}_{sys}$ = air systems output.
- $C_z \frac{dT_z}{dt}$ = energy stored in zone air.
- $C_z = \rho_{air} C_p C_T$, with ρ_{air} zone air density, C_p specific heat, C_T sensible-heat capacity multiplier.

In a single-zone residence, the interzone-mixing term vanishes. The surface-convection sum is condensed into the air-mass coupling $h^{\text{am}}(T^{\text{mass}} - T^{\text{air}})$ once all interior surfaces are lumped into one aggregate node with temperature T^{mass}. What remains (infiltration, internal gains, HVAC delivery, and the mass coupling) is precisely the air-node balance of Eq. (1). The mass-node balance of Eq. (2) then plays the role of the surface heat balance that EnergyPlus solves layer-by-layer with conduction transfer functions, lumped here into a single capacitance.

B.2 Mapping Between SSM and RC Notation

For readers familiar with structured state-space models (S4 [17], S5 [32], Mamba [16]), this appendix makes explicit the correspondence between the diagonal-scalar SSM recurrence and the NeuralBES two-node thermal recurrence.

A continuous-time linear SSM is

$$\frac{dx(t)}{dt} = Ax(t) + Bu(t), \qquad y(t) = Cx(t) + Du(t),$$

and its zero-order-hold discretization with step Δt is

$$x_t = \bar{A}\, x_{t-1} + \bar{B}\, u_t, \qquad \bar{A} = \exp(A\Delta t), \qquad \bar{B} = (A^{-1})(\bar{A} - I)B.$$

In the diagonal case (A scalar per state dimension), this collapses to

$$x_t = \bar{a}\, x_{t-1} + \bar{b}\, u_t, \qquad \bar{a} = e^{a\Delta t}.$$

The NeuralBES air-node update Eq. (4) has exactly this form with

$$a = -\frac{P^{\text{air}}}{C^{\text{air}}}, \qquad \bar{a} = \alpha^{\text{air}} = \exp(-P^{\text{air}}\Delta t / C^{\text{air}}), \qquad \bar{b}\, u_t = (1 - \alpha^{\text{air}})\, T^{\text{air,eq}}.$$

The full per-building state is two-dimensional $(T^{\text{air}}, T^{\text{mass}})$, and the coupling h^{am} appears in the off-diagonal of A; after ZOH discretization we apply one scalar scan per node and use the other node's current estimate as input, which is exactly the decoupled-scalar S5 parallelization strategy.

The key divergence from the SSM literature is that α^{air} is *not* a free learned parameter. In S4/S5/Mamba, $\bar{A}$ is learned directly (possibly via a HiPPO initialization or a selection mechanism). In NeuralBES, $\bar{A}$ is a *function of encoder outputs* via $\alpha = \exp(-P/C \cdot \Delta t)$, where P and C are positive-constrained MLP outputs conditioned on the static building descriptor. The parallel-scan machinery is identical; the parameterization of the state transition is not. This is what lets NeuralBES inherit the $O(\log T)$ training complexity of structured SSMs while keeping its $\bar{A}$ interpretable as physical conductances over capacitances.

Forcing term. The affine forcing $\bar{b}\, u_t$ in SSM notation carries the exogenous inputs (u_t) through an input projection. In NeuralBES, the affine term $\mathbf{b}^{\text{air}}_{b,t} = (1 - \alpha)\, T^{\text{air,eq}}$ is the *conductance-weighted sum of physical forcings* (HVAC load, internal gains, outdoor temperature, and mass temperature) scaled by the non-survival fraction of one timestep. It is built from encoder-predicted coefficients (UA^{infil}, h^{am}, etc.) multiplied by the time-varying exogenous signal $\mathbf{w}_{b,t}$, rather than a free linear layer.

Output projection. NeuralBES does not have a separate C, D output matrix: the state itself (T^{air}) is directly observed as one of the three target channels, and the other two ($Q^{\text{heat}}, Q^{\text{cool}}$) are recovered from the saturated $\dot{Q}^{\text{sys}}$ that was computed during the predictor step. In SSM language, $C = I$ for the temperature channel and the energy channels are read off the actuator-saturation branch rather than from a learned linear readout.

C Architecture and Implementation Details

This appendix consolidates implementation choices omitted from Section 3 for space: parameter-head widths and output scales, the chunked log-space scan, and state initialization.

Parameter heads. Each parameter head is a two-layer MLP of the form $\mathbf{z}_b \mapsto \mathrm{Linear}(\mathrm{SiLU}(\mathrm{Linear}(\mathbf{z}_b)))$. Hidden widths are $d/2$ for the thermal and HVAC heads and $d/4$ for the lighter-weight infiltration, solar, and initial-condition heads, where $d{=}512$ is the shared embedding dimension. The final linear layer of every head is zero-initialized so that training begins from a neutral physical prior. Each raw output is passed through a bounded activation with a head-specific scale to guarantee physically plausible ranges: capacitances $C^{\text{air}}, C^{\text{mass}}$ via $\mathrm{softplus}(\cdot) \cdot \{5{\times}10^5, 5{\times}10^6\}$ J/K respectively, conductances $h^{\text{am}}, UA^{\text{env}}$ via $\mathrm{softplus}(\cdot) \cdot \{200, 100\}$ W/K, heating and cooling capacities via $\mathrm{softplus}(\cdot) \cdot 5{,}000$ W, rated COPs via $\mathrm{softplus}(\cdot) + 1$ (guaranteeing COP$>$1), base ACH via $\mathrm{softplus}(\cdot) \cdot 0.5$, and wind coefficient via $\mathrm{softplus}(\cdot) \cdot 0.05$. The internal-gains head is time-aware: it takes the concatenation of $\mathbf{z}_b$ with the per-timestep schedule vector and outputs convective/radiative gain shares via a learned per-schedule scale (softplus, units W per unit schedule fraction) plus a learned base load.

Chunked log-space scan. The Heinsen log-space formulation of Section 3.3 proceeds as follows: (i) compute element-wise log-decays $\ell_{b,t} = \ln \alpha_{b,t}$; (ii) form cumulative sum $L_{b,t}$ and $\Lambda_{b,t} = e^{L_{b,t}}$; (iii) rescale forcings as $\tilde{\mathbf{b}}_{b,t} = \mathbf{b}_{b,t}\, e^{-L_{b,t}}$, expressing each in the common reference frame of the initial condition; (iv) form cumulative sum $S_{b,t}$; (v) recover $T_{b,t} = \Lambda_{b,t}(T_{b,0} + S_{b,t})$. To bound the growth of the rescaled forcings $e^{-L_{b,t}}$ over many timesteps, we apply the scan in chunks of length M: within each chunk the Heinsen scan is fully parallel, and only the scalar boundary state is carried sequentially between chunks. The chunk size is adaptive,

$$M = \max\left(4,\ \min\left(64,\ \left\lfloor \frac{80}{\max_{b,t}(-\ln \alpha_{b,t})} \right\rfloor\right)\right), \tag{9}$$

which keeps $e^{-L_{b,M}}$ well below the float32 maximum of roughly 3.4×10^{38} inside every chunk while keeping chunks long enough for the parallel scan to remain efficient.

Initial conditions. The inference-time offset of Section 3.4 is

$$T^{\text{air}}_{b,0} = T^{\text{sp,mid}}_{b,0} + \Delta^{(0)}_b, \qquad T^{\text{sp,mid}}_{b,0} = \tfrac{1}{2}(T^{\text{heat,sp}}_{b,0} + T^{\text{cool,sp}}_{b,0}), \tag{10}$$

where $\Delta^{(0)}_b$ is produced by a small two-layer MLP conditioned on $\mathbf{z}_b$ and the first-step forcing $\mathbf{w}_{b,0}$, and the mass temperature $T^{\text{mass}}_{b,0}$ is initialized identically. Because $\Delta^{(0)}_b$ would otherwise receive no gradient under teacher-forcing, we add a small auxiliary MSE loss matching the inference-style initialization to the ground-truth first-step temperature.

Warm-up to steady state. Because a parallel scan must commit to the initial state before advancing any timestep, a poorly chosen $T_{b,0}$ can inject a transient that contaminates the first several hours of the rollout. To remove this dependence, we prepend $W = 16$ warm-up timesteps (four hours at $\Delta t = 900\,\text{s}$) that replicate the first-step forcing $\mathbf{w}_{b,0}$ and setpoints; the scan is executed over the concatenated horizon of length $T + W$, and only the trailing T outputs are retained as predictions. Under constant inputs the two-node LDS is contractive, so the state trajectory relaxes exponentially toward the quasi-steady equilibrium defined by those forcings before the real driver sequence begins. This mirrors the implicit warm-up performed by sequential simulators such as EnergyPlus and removes an otherwise significant source of bias on the first day of the horizon, at negligible cost because the extra W steps are absorbed into the same parallel scan.

D Dynamic Diagnostics

This appendix collects three dynamic diagnostics referenced in Section 4.2: the per-day error trajectory over the weekly rollouts (Figure 4), the ramp-rate distribution of ΔT_{in} per 15 min (Figure 5), and the outdoor-temperature-stratified MAE improvement of NeuralBES over each baseline (Figure 6). These figures support the narrative of Section 4.2 but are not themselves the headline claims, so they are moved here for space.

Ramp-rate fidelity. A thermal emulator that predicts the setpoint midpoint will score well on point-error metrics but badly on temporal-derivative metrics, which are what matter for grid-interactive HVAC and demand response. Figure 5 shows the distribution of ΔT_{in} per 15 min against ground truth. The actual distribution has a sharp central spike at zero and heavy tails spanning roughly four decades. NeuralBES tracks this shape most closely across the full $\pm 0.5\,^{\circ}$C range; LSTM, Roformer,

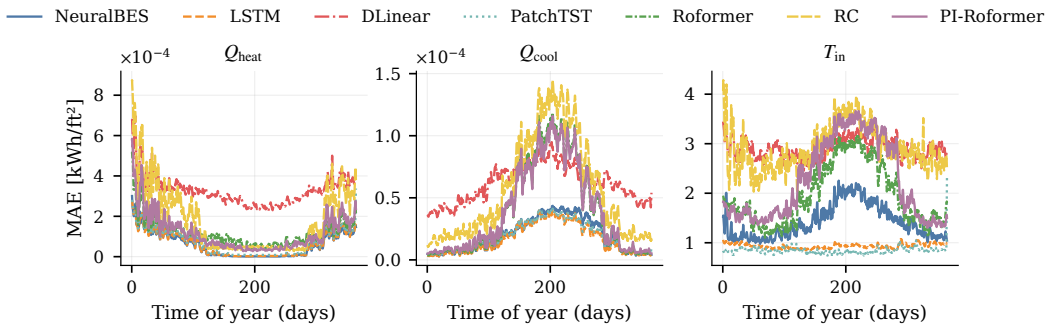


Figure 4: Per-day MAE on the three-state (multi-region) weekly rollouts, aggregated across 672-step windows and stratified by target channel. Q_{heat} (left) errors concentrate in winter (low T_{out}); Q_{cool} (middle) errors bulge in summer (days 150–240); T_{in} (right) errors follow the seasonal envelope in which the envelope–infiltration dynamics are most stressed. NeuralBES stays in the bottom tier of curves for both HVAC channels: flat and low on Q_{heat}, and matching PatchTST and LSTM near the summer Q_{cool} peak. DLinear's Q_{heat} error is elevated across seasons and RC's Q_{cool} peaks sharply during the summer maximum, consistent with the seasonal rollout trajectories in Figure 3 (main text).

and PI-Roformer recover the central spike but with tails that are slightly broader than the truth. The two extreme failure modes are instructive. RC produces a conspicuously *over-wide* distribution (its tails sit almost an order of magnitude above the actual density at $\pm 0.4\,^{\circ}$C) because the fixed-structure 3R2C rollout without COP derating or learned thermostat latency responds too abruptly to forcing changes. DLinear produces a flat distribution with a muted central peak, over-predicting ramp events by more than an order of magnitude in the tails while under-predicting the density of near-zero hold periods. This explains the paradox that DLinear's T_{in} RMSE is only modestly worse than NeuralBES despite being unusable for control.

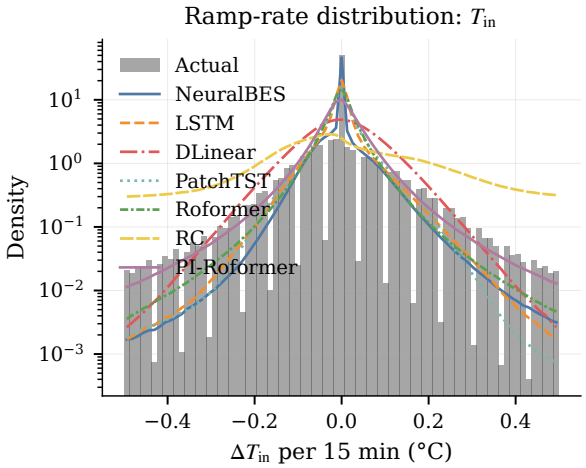


Figure 5: Ramp-rate distribution of ΔT_{in} per 15 min (log density) on the multi-region weekly rollouts. NeuralBES matches the central spike and the heavy tails of ResStock ground truth; RC and DLinear produce over-wide distributions that over-predict large ramps; LSTM, Roformer, and PI-Roformer recover the central spike with slightly broader tails.

Extreme-weather regime. Figure 6 stratifies the MAE improvement of NeuralBES by outdoor temperature, and the per-channel picture is sharply different. DLinear's Q_{heat} error explodes when $T_{\text{out}} > 25\,^{\circ}$C because a linear time-mixer cannot output zero in a regime where heating is structurally off. On Q_{cool}, the largest gap is in the shoulder regime ($T_{\text{out}} \approx$ 10–15 °C) where cooling is spooling up but training density is low, rather than in the hot tail where NeuralBES, LSTM, and PatchTST converge. On T_{in}, DLinear and RC are weakest at low T_{out}, consistent with their ramp-rate miscalibration above; PatchTST is flat across the entire T_{out} range because its indoor-temperature prediction is insensitive to outdoor temperature at all, as Fig. 2 confirms. RC under-predicts extreme loads because its fixed COP over-estimates equipment effectiveness at extreme outdoor temperatures.

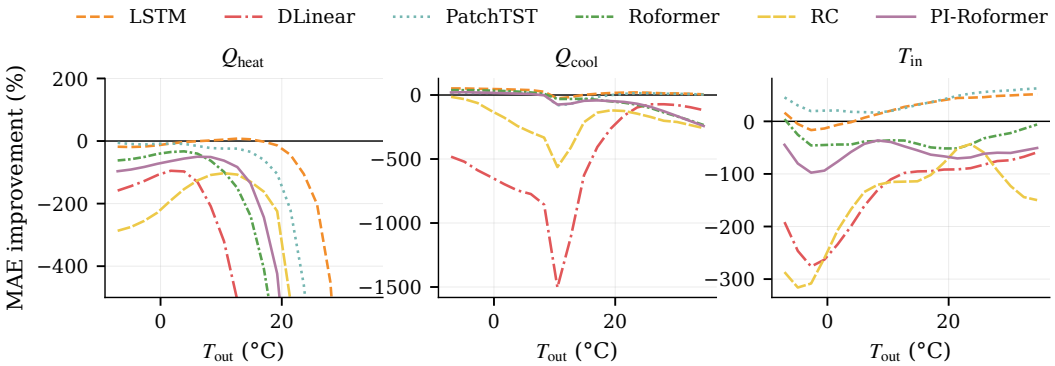


Figure 6: MAE improvement of NeuralBES over each baseline (in percent of baseline MAE), stratified by outdoor temperature T_{out}, on the multi-region test set. Negative values indicate NeuralBES is better; the magnitude is the relative error reduction. Note the different vertical scales: Q_{heat} is in units of $10^6\%$ because DLinear's error blows up when $T_{\text{out}} > 25\,^\circ\text{C}$, Q_{cool} peaks around -1500% for DLinear near $T_{\text{out}} = 10\,^\circ\text{C}$, and T_{in} degradation is largest for DLinear and RC in cold regimes ($T_{\text{out}} < 0\,^\circ\text{C}$).

E Regional Weekly-Rollout Evaluation

This appendix reports the per-region weekly-rollout numbers behind the multi-region aggregate in Table 2, the results of the larger LSTM, and a leave-one-region-out transfer experiment. Each region (CA, NY, TX, Multi) is evaluated on the 672-step (seven-day) weekly protocol described in Section 4.1.

Capacity-matched and large LSTM. Table 2 reports the capacity-matched LSTM (0.98 M parameters). Table 3 reports, under the same annual protocol, the larger 36.6 M LSTM, which served as the LSTM baseline in the submitted version and is the LSTM shown in Figures 2–7 and Table 4. Reducing the LSTM to the size of NeuralBES lowers its error on all three channels, so the accuracy advantage of the LSTM does not stem from its capacity; at either size, it satisfies neither mode exclusivity nor bounds adherence.

Model	Params (M)	MAPE (%) Q_{heat}	MAPE (%) Q_{cool}	RMSE (°C) T_{in}	Mode excl. [%]	Bounds [%]
LSTM (large)	36.6	$38.5_{\pm 0.1}$	$37.6_{\pm 1.7}$	$1.48_{\pm 0.01}$	$36.9_{\pm 18.0}$	$63.1_{\pm 18.0}$
LSTM (capacity-matched)	0.98	$32.6_{\pm 2.4}$	$30.9_{\pm 4.9}$	$1.30_{\pm 0.10}$	$32.4_{\pm 9.6}$	$67.2_{\pm 9.4}$
NeuralBES	1.0	$41.5_{\pm 4.3}$	$41.3_{\pm 0.4}$	$2.79_{\pm 0.03}$	$100.0_{\pm 0.0}$	$99.7_{\pm 0.1}$

Table 3: Annual full-year (35,040-step) rollout on the multi-region ResStock test set for the two LSTM sizes, with NeuralBES for reference. The protocol and metrics are those of Table 2.

Transfer to an unseen region. We also train NeuralBES on two regions and evaluate it on the third, whose climate-zone categories are absent from training. The temperature RMSE increases by at most $0.12\,^\circ\text{C}$ (and decreases for NY and TX), whereas the load MAPE increases by 3–10 pp, indicating that envelope dynamics transfer more readily than load magnitudes. This experiment covers NeuralBES only, so we make no comparative claim.

Horizon-degradation analysis. Reading Table 4 against the annual Tables 2 and 3 separates architectures that compound error across 35,040 steps from those that do not. Roformer collapses (Q_{heat} MAPE $35.4 \to 67.8\%$, Q_{cool} $32.5 \to 85.3\%$, T_{in} RMSE $1.38 \to 3.26\,^\circ\text{C}$); NeuralBES degrades by $\times 1.10$–1.30 on the same three metrics while the 36.6 M LSTM is essentially flat. NeuralBES's stability is structural: $\alpha = \exp(-P\Delta t/C) \in (0, 1)$ (Section 3.3) makes the discrete-time update a contraction, so transient errors decay rather than compound.

Region	Metric		DLinear	LSTM	PatchTST	Roformer	PI-Roformer	RC	NeuralBES
CA	MAPE (%)	Q_{heat}	192.65	88.50	130.74	67.80	**63.91**	91.20	66.75
		Q_{cool}	258.79	58.43	94.41	49.25	**45.62**	72.44	50.45
	RMSE (°C)	T_{in}	2.94	2.01	2.09	1.70	**1.62**	2.66	2.63
NY	MAPE (%)	Q_{heat}	76.46	34.38	34.01	**30.67**	35.19	97.82	30.68
		Q_{cool}	119.41	46.45	50.00	**40.20**	41.18	107.96	42.54
	RMSE (°C)	T_{in}	2.62	1.76	1.43	**1.36**	1.61	2.81	1.98
TX	MAPE (%)	Q_{heat}	226.07	54.58	79.06	42.50	42.82	114.25	**41.17**
		Q_{cool}	94.54	34.07	38.25	32.26	30.94	95.62	**29.58**
	RMSE (°C)	T_{in}	2.23	1.66	1.56	**1.29**	1.31	3.08	1.93
Multi	MAPE (%)	Q_{heat}	195.23	38.45	43.57	**35.37**	41.68	107.51	37.05
		Q_{cool}	132.96	35.56	40.12	**32.51**	39.54	120.55	37.53
	RMSE (°C)	T_{in}	3.72	1.46	1.59	**1.38**	1.63	4.38	2.15
Physics (Multi)	Mode excl. [%]		8.44	17.72	36.20	43.38	2.75	100.00	**100.00**
	Bounds [%]		91.31	82.23	63.57	56.53	96.10	**98.69**	98.50

Table 4: Weekly-rollout (672-step) comparison on the ResStock test buildings. The top block reports per-target errors stratified by region: Q_{heat} and Q_{cool} as MAPE (%), T_{in} as RMSE (°C). The bottom block reports *mode exclusivity* and *bounds adherence* (defined in Section 4.2) on the Multi rollout. LSTM is the 36.6M model.

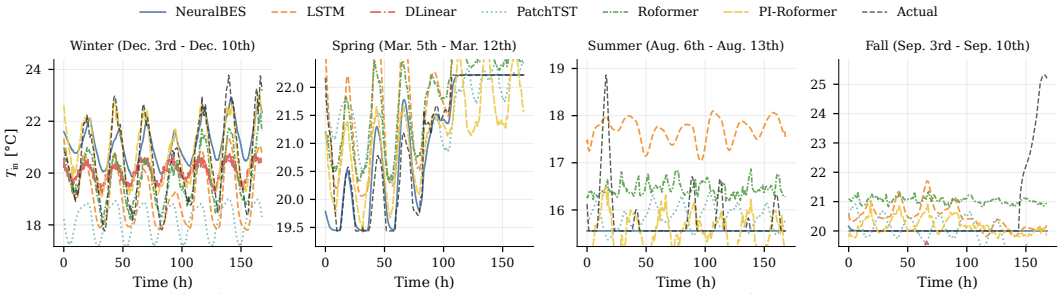


Figure 7: Specific failure-mode case study: two-week T_{in} rollouts for a single multi-region test building across four seasons. In the fall panel, the ground truth (black dashed) swings between ~ 22 and $\sim 26\,^{\circ}\text{C}$ following a setpoint excursion, while Roformer (green) and PI-Roformer (yellow) sit flat at $\sim 21\,^{\circ}\text{C}$ and fail to track the thermal interaction at all. In summer, Roformer drops to $\sim 16\,^{\circ}\text{C}$ and stays flat while the ground truth oscillates between ~ 16 and $\sim 18\,^{\circ}\text{C}$; only NeuralBES (blue) reproduces the diurnal swing. Winter and spring panels show all models behave reasonably: the failure is regime-specific to seasons in which envelope-driven dynamics dominate.

F Experimental Setup

This appendix consolidates the configuration of every baseline and the compute environment under which the experiments of Table 2 and Table 4 were run. Section F.1 summarizes the architectural family and static-injection mechanism of each baseline and reports the corresponding hyperparameters; Section F.2 describes the hardware and wall-clock budget.

F.1 Baseline Architectures and Hyperparameters

Table 5 provides the six baselines spanning the inductive-bias spectrum while Table 6 reports the architectural and optimization hyperparameters used for every baseline in Table 2. All baselines share the same input interface (weather, schedules, time encodings, and static features through the shared static encoder), the same train/val/test split (0.7/0.15/0.15 over 2000/428/428 buildings per state), the same 672-step sequence length, the same 200-epoch budget with patience 10, and the same Smooth L1 loss with $\beta = 5\times10^{-3}$. Differences across rows reflect the capacity regime typical of each family rather than tuning.

Model	Family	Static injection	Key architectural choice
DLinear [40]	linear	additive bias	trend/seasonal decomposition
LSTM	recurrent	FiLM (id-init)	width 144 (0.98 M) or 512 (36.6 M)
PatchTST [24]	transformer	FiLM (id-init)	patching 48/24, RevIN, channel-indep.
Roformer [33]	transformer	FiLM (id-init)	encoder–decoder, RoPE
PI-Roformer	hybrid	FiLM (id-init)	Roformer + soft RC heat-balance residual
RC Grey-Box	physics	encoder-predicted params	3R2C w/ fixed COPs, fixed setpoint band
NeuralBES	**physics-encoded SSM**	**encoder-predicted params**	**parallel log-space scan + pred.–corr.**

Table 5: Baselines compared in this paper. PI-Roformer tests whether a shallow trunk-plus-physics-head hybrid suffices or whether the full NeuralBES integration is needed.

Model	Hidden d	Layers	Heads	Dropout	Batch	LR
DLinear	512	1	—	0.0	512	1.0×10^{-4}
LSTM (36.6 M)	512	6	—	0.2	256	1.0×10^{-4}
LSTM (0.98 M)	144	2	—	0.2	256	1.0×10^{-4}
PatchTST	128	3	4	0.1	128	1.0×10^{-4}
Roformer	512	6	8	0.2	128	1.0×10^{-4}
PI-Roformer	512	3	8	0.0	256	1.0×10^{-4}
RC Grey-Box	128	1	—	0.0	512	1.0×10^{-4}
NeuralBES	**512**	**—**	**—**	**0.0**	**512**	$\mathbf{1.0\times10^{-4}}$

Table 6: Baseline hyperparameters for the multi-region runs. PatchTST uses patch length 48 (12 h) and stride 24 (6 h) on the 15-minute series. NeuralBES has no transformer/recurrent depth in the conventional sense: d=512 is the shared embedding dimension produced by the static encoder and consumed by the parameter heads of Appendix C. All runs use AdamW with weight decay 10^{-2}, gradient clipping at norm 1.0, and a 2000-step linear LR warmup.

F.2 Compute Resources

All training and evaluation jobs ran on a single-node Slurm cluster with one NVIDIA L40S GPU (48 GB VRAM) and 16 CPU cores per job. Each baseline is trained for up to 200 epochs with early stopping (patience 10) under a fixed wall-clock budget; in practice runs converge within roughly 15 minutes of wall-clock training time for the 2000-buildings-per-state multi-region setup at 15-minute resolution and 672-step weekly windows. Memory footprint stays under 24 GB of GPU memory for every baseline at the batch sizes in Table 6, including the 36.6 M-parameter LSTM. The total compute reported in the main text aggregates seven model families across three regions plus a multi-region split, with three seeds for the entries reporting variance in Table 2; preliminary and ablation runs not reported here multiply this by approximately a factor of three.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The four contributions stated in the abstract and Section 1 (neural parameterization of the two-node RC model, log-space parallel scan, predictor–corrector thermostat loop, and zero-shot prediction for held-out buildings with a single shared encoder) are the claims supported in Section 3 and the empirical results of Section 4.2 (Table 2, Figures 2 and 3).

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Section 5 discusses two limitations explicitly which are the absence of a domestic hot-water sub-model and the rigidity of the governing ODE when extending to multi-zone, refrigerant, or stratified-storage phenomena. Appendix A additionally notes that we model only sensible (not latent) HVAC loads, which under-states cooling electricity in humid climates, a limitation shared with the 3R2C literature. Section 5 further notes that the model is evaluated only against simulated data with known exogenous inputs, that throughput at stock scale is not benchmarked, that the learned RC coefficients are effective rather than measured parameters, that the thermostat model is restricted to single-stage equipment, and that results are stratified by region but not by building type.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.

- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: The paper does not state numbered theorems, but the closed-form solution of the per-step linear ODE (Eq. (4)) is derived under an explicitly stated piecewise-constant-forcing assumption in Section 3, and the SSM/RC correspondence and ZOH discretization are derived in full in Appendix B.2. The reduction from the full EnergyPlus zone-air heat balance to the two-node form of Eqs. (1)–(2) is given in Appendix B.1.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Section 3 fully specifies the architecture (two-node RC parameterization, parallel log-space scan, three-iteration predictor–corrector loop), and Appendix C gives the per-head MLP widths, bounded-activation scales, chunked-scan rule, warm-up window, and initial-condition heads. The data splits, sequence length, baseline configurations, and evaluation protocols are specified in Section 4.1.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.

- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The code, including the script to gather the public dataset, is attached in the supplement materials to reproduce the result. The final codebase will be available upon request.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Section 4.1 specifies the train/val/test split (0.7/0.15/0.15, 2000/428/428 buildings per state) and the 672-step weekly evaluation protocol, and gives the baseline configurations (Table 5); Appendix C specifies the architectural hyperparameters (hidden width d=512, head widths, activation scales, chunked-scan size M, warm-up steps W=16, N_{iter}=3 predictor–corrector passes).

   Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Table 2 reports mean $\pm$ standard deviation over three seeds for every model and every metric (MAPE on Q_{heat} and Q_{cool}, RMSE on T_{in}, mode exclusivity, and bounds adherence), each seed varying initialization and data-shuffle order under the train/val/test split of Section 4.1.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
   - If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Appendix F.2 reports the compute setup: a single-node Slurm cluster with one NVIDIA L40S GPU (48 GB VRAM) and 16 CPU cores per job, with each run completing in roughly 15 minutes of wall-clock training time and staying under 24 GB of GPU memory at the batch sizes in Table 6. The architectural budget — d=512 shared embedding, parameter counts of all models in Table 2 (NeuralBES at 1.0 M), and the $O(\log T)$-depth parallel scan over T=35,040 steps described in Section 3 — bounds the compute requirement.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The work uses only synthetic, simulator-generated building energy traces from the publicly released ResStock corpus (Section 4.1) and involves no human subjects, personally identifiable information, or scraped content; the methodology in Section 3 and the evaluation in Section 4.2 conform with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Section 1 motivates the work by its positive impact on grid decarbonization and demand-side flexibility for residential building stocks, where scalable physics-valid emulators support utility-scale planning and policy modeling. Negative impacts are limited because the model emulates synthetic ResStock outputs (not metered reality, as scoped in Appendix A) and contains no personally identifiable building information; misuse risks are correspondingly low.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

Justification: NeuralBES is a regression emulator for residential thermal dynamics trained on synthetic ResStock simulations (Section 4.1) and does not produce content (text, images, code) of the kind that the safeguards guideline targets; it carries no high-risk dual-use surface.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: The ResStock dataset [28] and the EnergyPlus simulator [8] that generated it are cited in Section 4.1 and Section 1; baseline architectures (DLinear, PatchTST, Roformer, S4, Mamba, LRU, LSTM) are cited at point of use in Section 1 and Section 4.1 (Table 5). All cited assets are publicly released under their respective open licenses and are used consistently with their terms.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [N/A]

Justification: The paper does not release new datasets or pre-trained model checkpoints.

Guidelines:

- The answer [N/A] means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.

- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The work involves no crowdsourcing or human-subjects research; all training and evaluation data are synthetic ResStock simulations as documented in Section 4.1.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The work involves no human subjects, as scoped in Section 4.1.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: The presented research does not involve LLMs as an important component

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.