Available online at www.sciencedirect.com

ScienceDirect

IFAC PapersOnLine 56-2 (2023) 5855–5860

Physically Consistent Neural ODEs for Learning Multi-Physics Systems ⋆

M. Zakwan† * L. Di Natale† **,* B. Svetozarevic ** P. Heer **
C. N. Jones * G. Ferrari Trecate *

* *Laboratoire d'Automatique, EPFL, Lausanne, Switzerland.*
** *Urban Energy Systems Laboratory, Empa, Dübendorf, Switzerland.*

Abstract: Despite the immense success of neural networks in modeling system dynamics from data, they often remain physics-agnostic black boxes. In the particular case of physical systems, they might consequently make physically inconsistent predictions, which makes them unreliable in practice. In this paper, we leverage the framework of Irreversible port-Hamiltonian Systems (IPHS), which can describe most multi-physics systems, and rely on Neural Ordinary Differential Equations (NODEs) to learn their parameters from data. Since IPHS models are consistent with the first and second principles of thermodynamics by design, so are the proposed Physically Consistent NODEs (PC-NODEs). Furthermore, the NODE training procedure allows us to seamlessly incorporate prior knowledge of the system properties in the learned dynamics. We demonstrate the effectiveness of the proposed method by learning the thermodynamics of a building from the real-world measurements and the dynamics of a simulated gas-piston system. Thanks to the modularity and flexibility of the IPHS framework, PC-NODEs can be extended to learn physically consistent models of multi-physics distributed systems.

Copyright © 2023 The Authors. This is an open access article under the CC BY-NC-ND license (https://creativecommons.org/licenses/by-nc-nd/4.0/)

Keywords: Machine Learning, Neural networks, Multi-physics, Thermodynamics, Data-driven Modelling, Irreversible port-Hamiltonian systems.

1. INTRODUCTION

In recent years, Neural Networks (NNs) have achieved impressive performances on a broad range of tasks, including time series prediction, where Recurrent NNs (RNNs), Gated Recurrent Units (GRUs), Long Short-Term Memory networks (LSTMs), and transformers, often attain great accuracy (Wang and Yu, 2021). These successes also motivated researchers to use NNs to identify system dynamics from data, but such models often suffer from physical inconsistencies: they can fit data well without learning the underlying ground truth, making them unreliable in practice (Geirhos et al., 2020; Di Natale et al., 2022). As a countermeasure to the brittleness of NN-based models, there has been increasing interest in incorporating prior knowledge – also known as inductive bias – into NNs to ensure physical consistency, leading to Hamiltonian NNs (Chen et al., 2019; Greydanus et al., 2019; Finzi et al., 2020), Lagrangian NNs (Cranmer et al., 2020), or Poisson NNs (Jin et al., 2022), amongst others. We defer the reader to Wang and Yu (2021) for a comprehensive survey on physics-guided deep learning for dynamical systems.

At the same time, higher-level connections between NNs and dynamical systems have also been studied, showing that some classes of NNs can be interpreted as discretized dynamical systems (Haber and Ruthotto, 2017). On the other hand, Chen et al. (2018) proposed the framework of Neural Ordinary Differential Equations (NODEs), where inputs are transformed through a continuous-time ODE embedding trainable parameters. In other words, NODEs learn the parameters of an ODE to fit data, making them particularly suitable to model complex dynamical systems (Greydanus et al., 2019; Rubanova et al., 2019). Furthermore, their interpretation as ODEs allows one to borrow tools from dynamical system theory to analyze their properties (Zakwan et al., 2022; Fazlyab et al., 2022; Galimberti et al., 2021). However, similarly to classical NNs, NODEs can be physically inconsistent in general.

This paper proposes Physically Consistent NODEs (PC-NODEs), which leverage the Irreversible port-Hamiltonian (IPH) modeling framework to describe multi-physics systems and NODEs to learn their dynamics. Thanks to the IPH formulation, we can guarantee that PC-NODEs respect the first and second laws of thermodynamics at all times and by construction, solving the issue of physically inconsistent NODEs. Moreover, unlike black-box NNs, PC-NODEs allow us to embed *a priori* desired structural properties of trainable parameters, such as skew-symmetry and prescribed sparsity patterns, in the learning process.

Our efforts to ground learning schemes in the underlying physics are conceptually related to the work of Masi et al. (2021), who ensured that suitable model derivatives are consistent with the rules of thermodynamics. In another attempt, Di Natale et al. (2022) introduced Physically Consistent NNs (PCNNs), where a physics-inspired module runs in parallel to a NN to ensure the predictions comply with underlying physical laws. While these methods

⋆ This research was supported by the Swiss National Science Foundation under the NCCR Automation (grant agreement 51NF40_180545) and in part by the Swiss Data Science Center (grant no. C20-13). † The authors contributed equally. Corresponding authors: muhammad.zakwan@epfl.ch, loris.dinatale@empa.ch.

2405-8963 Copyright © 2023 The Authors. This is an open access article under the CC BY-NC-ND license.
Peer review under responsibility of International Federation of Automatic Control.
10.1016/j.ifacol.2023.10.079

were shown to work well in case studies, they are limited to a few applications. In contrast, PC-NODEs are more general and applicable to a wide variety of systems.

The modularity of the IPH framework allows us to characterize many multi-physics systems, including thermodynamic, mechanical, chemical, or electrical systems (Ramirez et al., 2013a; Van der Schaft and Jeltsema, 2021). Furthermore, identifying system dynamics in the IPH form provides several benefits, as one can then design stabilizing controllers and scale to distributed systems via interconnection with other passive port-Hamiltonian systems (Ramirez et al., 2013a). To showcase the flexibility of the proposed PC-NODEs, in this paper, we model the thermal dynamics of a building from the real-world measurements and the dynamics of a simulated gas-piston system.

Organization: Section 2 presents PC-NODEs and describes the training procedure. In Section 3, we consider the modeling of two case studies, and the results are illustrated in Section 4. Finally, Section 5 concludes the paper.

Notations: The $p-$norm is denoted as $||\cdot||_p$. A matrix J is *skew-symmetric* if $J = -J^\top$. The *Poisson bracket* of $Z, G \in \mathcal{C}^\infty(\mathbb{R}^n)$ with respect to a skew-symmetric matrix J is defined as $\{Z, G\}_J = \frac{\partial Z^\top(x)}{\partial x} J \frac{\partial G(x)}{\partial x}$.

2. LEARNING IRREVERSIBLE PORT-HAMILTONIAN DYNAMICS

This section introduces the PC-NODE framework to learn system dynamics from data while ensuring compatibility with the first and second laws of thermodynamics.

2.1 Physics-Consistent NODEs

An IPH system (Ramirez et al., 2013a,b) is described as

$$\dot{x} = R\left(x, \frac{\partial H(x)}{\partial x}, \frac{\partial S(x)}{\partial x}\right) J \frac{\partial H(x)}{\partial x} + W\left(x, \frac{\partial H(x)}{\partial x}\right) + g\left(x, \frac{\partial H(x)}{\partial x}\right) u \,, \quad (1)$$

where $x \in \mathbb{R}^n$ is the state, $u \in \mathbb{R}^m$ the control input, and the different functions and matrices satisfy [1]

- (P_1) the Hamiltonian function H and the entropy function S are maps from $\mathcal{C}^\infty(\mathbb{R}^n)$ to $\to \mathbb{R}$;
- (P_2) the interconnection matrix $J \in \mathbb{R}^{n\times n}$ is constant and skew-symmetric;
- (P_3) the real function $R = R(x, \frac{\partial H}{\partial x}, \frac{\partial S}{\partial x})$ is defined as
$$R\left(x, \frac{\partial H}{\partial x}, \frac{\partial S}{\partial x}\right) = \gamma\left(x, \frac{\partial H}{\partial x}\right)\{S, H\}_J \,, \quad (2)$$
where $\gamma \succeq 0$ is a nonnegative function of the states and co-states of the system;
- (P_4) the two vector fields W and g satisfy $W(x, \frac{\partial H}{\partial x}) \in \mathbb{R}^n$ and $g(x, \frac{\partial H}{\partial x}) \in \mathbb{R}^{n\times m}$.

We have used the **blue color** to denote functions that can be parameterized, e.g. using NNs, and identified from data as described in Section 2.2. This defines the overall PC-NODE framework. As long as the learned parameters respect the constraints and properties listed above, the learned model will obey the first and second laws of thermodynamics by construction. Indeed, by the skew-symmetry of J, setting $W, u \equiv 0$, we have

$$\frac{dH}{dt} = \frac{\partial H^\top}{\partial x}\left(RJ\frac{\partial H}{\partial x}\right) = R \times \left(\frac{\partial H^\top}{\partial x} J \frac{\partial H}{\partial x}\right) = 0 \,, \quad (3)$$

which proves the conservation of energy in the system. Similarly, we can show the irreversible creation of entropy in the system as follows:

$$\frac{dS}{dt} = R\frac{\partial S^\top}{\partial x} J \frac{\partial H}{\partial x} = \gamma\left(x, \frac{\partial H}{\partial x}\right)\{S, H\}_J^2 \geq 0 \,,$$

as long as $\gamma \succeq 0$. We defer the reader to Ramirez et al. (2013b) for more details on these computations.

2.2 Training PC-NODEs

Several NODE training procedures have been proposed in the literature, such as the adjoint sensitivity method (Chen et al., 2018) or the auto-differentiation technique (Paszke et al., 2017). In this work, inspired by Haber and Ruthotto (2017), we first discretize PC-NODE (1) using the Forward-Euler (FE) method with sampling period $h > 0$, leading to

$$x_{i+1} = x_i + h\left(RJ\frac{\partial H(x_i)}{\partial x_i} + W + gu_i\right), \quad (4)$$

where x_i and x_{i+1} represent the current and next state, respectively. In practice, the step-size h is chosen sufficiently small so as to interpret the states in (4) as a sampled version of the state $x(t)$ of system (1).

We then assume to have access to a dataset of M sampled full-state trajectories

$$\mathcal{D} := \left\{(z_0^j, r_0^j), (z_1^j, r_1^j), \cdots (z_L^j, r_L^j)\right\}_{j=1}^M \,,$$

where L is the total number of time steps for each trajectory of measured states z and inputs r. Finally, we train system (4) to minimize the following objective

$$\min_{R,J,W,g} \sum_{i=1}^{L}\sum_{j=1}^{M} \ell(z_i^j, x_i^j) \,. \quad (5)$$

While we optimize the squared error $\ell(z, x) = ||z - x||_2^2$ in this work, this can easily be replaced by other loss functions.

We implement the proposed PC-NODEs using `PyTorch`, which allows us to easily propagate the inputs through the NODE and then rely on automatic BackPropagation Through Time (BPTT) (Werbos, 1990) to run Gradient Descent (GD) on the trainable parameters, as sketched in Fig. 1. However, in general, it does not allow one to introduce constraints on the parameters directly. In particular, it cannot not guarantee that either J satisfies property (P_2) or R (P_3). Consequently, the next Section discusses how to ensure that these constraints are satisfied – despite running unconstrained GD – in the light of two illustrative examples. While this may seem counter-intuitive at first, running unconstrained GD while enforcing constraints by construction allows one to leverage the full strength of automatic GD in `PyTorch`, which naturally scales to large systems and long prediction horizons.

[1] To have concise notation throughout the paper, the dependence on x and partial derivatives is dropped when it is clear from the context.

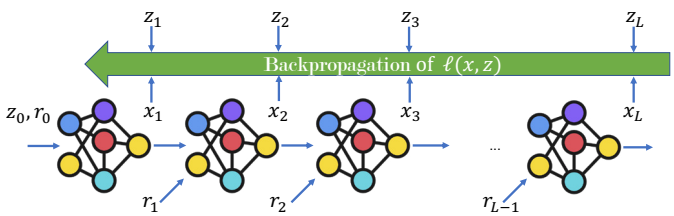

Fig. 1. A pictorial description of the training procedure of the discretized PC-NODE (4), where the same parameters are used at each step to propagate the state forward in time.

Remark 1. Besides modifying the loss function ℓ, one can also introduce weighted penalty terms in equation (5), e.g. to promote sparse solutions with $||J||_1$.

3. MODEL FORMULATIONS

To demonstrate the variety of systems that can be represented with IPH dynamics, this section describes how to model thermal building dynamics and gas-piston systems and ensure properties (P_1)–(P_4) are respected.

3.1 Thermal building dynamics

The thermal dynamics of a building can be seen as N connected thermal zones exchanging energy among themselves and with the outside, as depicted in Fig. 2. In this work, we assume that they are additionally impacted by various heat gains from heating or cooling operations and solar irradiation. Inspired by the IPH formulation of heat exchangers (Ramirez et al., 2013a), we model the entropy $S \in \mathbb{R}^N$ in each zone as follows

$$\dot{S} = \tilde{J}(T)\frac{\partial H(S)}{\partial S} + B_e(T)T_e + [B_s \;\; B_h \;\; B_c]\begin{bmatrix} Q_s \\ Q_h \\ Q_c \end{bmatrix}, \quad (6)$$

where $T \in \mathbb{R}^N$ represents the temperature in each zone. We separated the different inputs u, with $T_e \in \mathbb{R}$ corresponding to the ambient temperature, and Q_s, Q_h, $Q_c \in \mathbb{R}^N$ to solar, heating, and cooling gains for each zone, respectively. B_s, B_h, and B_c are $N \times N$ diagonal matrices gathering trainable parameters reflecting the impact of these gains on the entropy of each zone. $B_e(T) \in \mathbb{R}^N$ models the heat losses to the outside, with entries

$$B_e(T)_i = \lambda_{ie}\frac{(T_e - T_i)}{(T_i T_e)}$$

for each zone i, where $\{\lambda_{ie}\}_{i=1}^N$ are the trainable parameters. Finally, the skew-symmetric matrix $\tilde{J}(T) \in \mathbb{R}^{N\times N}$, lumping together R and J in this case, is parametrized as

$$\tilde{J}_{ij}(T) = -\tilde{J}_{ji}(T) = \begin{cases} \lambda_{ij}\frac{(T_j - T_i)}{(T_i T_j)} & \text{if } i \text{ is adjacent to } j \\ 0 & \text{otherwise,} \end{cases}$$

where two zones are *adjacent* if they share at least a common wall.

Interestingly, by the definition of entropy, and recalling that the Hamiltonian H represents the energy of the system, we have $\frac{\partial H(S)}{\partial S} = T$. Hence, there is no need to parametrize the partial derivatives of the Hamiltonian function in this case since they can be computed explicitly from the state of the system, as shown in Appendix A (assuming a constant volume for each zone).

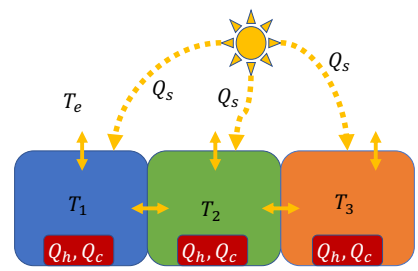

Fig. 2. A pictorial description of the thermal behavior of a three-zone building, where yellow arrows represent energy flows.

Proposition 1. (Consistency, and monotonicity). The PC-NODE (6) is consistent with the first and second laws of thermodynamics and monotonic with respect to all inputs, i.e. T_e, Q_s, Q_h, and Q_c if the learned parameters satisfy

$$B_s, B_h, B_c \succeq 0, \quad \text{and} \quad \lambda_{ij}, \lambda_{ie} \in \mathbb{R}_+, \quad \forall i, j = 1, \ldots, N.$$

Proof. See Appendix B for a sketch of the proof and Van der Schaft and Jeltsema (2021) for more details. □

Remark 2. The dependence of $\tilde{J}$ on T in (6) violates property (P_2). While state-dependent connection matrices break the consistency of the system with the first and second laws of thermodynamics in general (Ramirez et al., 2013a), we show that PC-NODE (6) remains consistent in the proof of Proposition 1.

Remark 3. Exploiting the linearity of the PC-NODE (6), one can show that it is almost equivalent to well-known Resistance-Capacitance (RC) architectures. The latter model the energy of each zone instead of their entropy, but both quantities are linked by definition since $dS = \frac{dH}{T}$. Multiplying (6) by the temperature of each zone, one can hence recover an energy model of the building, but with the training parameters in $B_s(T)$, $B_h(T)$, and $B_c(T)$ depending on the corresponding zone temperatures. Since the latter can be considered as roughly constant (in Kelvin), this is indeed similar to classical RC models.

3.2 Gas Piston system

Consider a typical gas piston system, as depicted in Fig. 3, where the piston is subject to friction, influenced by an external force $F(t) = u$, and its elasticity is modeled by a spring. We define the state of the system as $x = [S, V, q, p]^\top$, where S is entropy and V the volume of the gas, and q, p are the position and momentum of the piston, respectively. Inspired by Ramirez et al. (2013b), the system can be described by the following nonlinear IPH dynamics

$$\dot{x} = \left[R\left(x, \frac{\partial S}{\partial x}, \frac{\partial H(x)}{\partial x}\right)J_0 + J_1\right]\frac{\partial H(x)}{\partial x} + Gu\,, \quad (7)$$

$$J_0 = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 \end{bmatrix}, \qquad J_1 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \alpha \\ 0 & 0 & 0 & \beta \\ 0 & -\alpha & -\beta & 0 \end{bmatrix},$$

$$\frac{\partial H(x)}{\partial x} = [T, -P, Kq, v]^\top, \qquad G = [0, 0, 0, 1]^\top,$$

where $R(x, \frac{\partial S}{\partial x}, \frac{\partial H(x)}{\partial x}) = \frac{\mu v}{T}$, T is the temperature and P the pressure of the gas, K the spring constant, and $v = \frac{p}{m}$

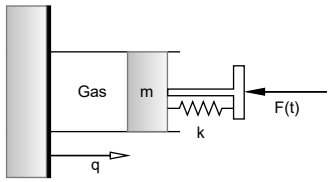

Fig. 3. Sketch of the gas piston system.

represents the speed of the piston with mass m and friction coefficient μ.

Since the entropy is a state of the system, $\frac{\partial S}{\partial x} = [1, 0, 0, 0]^\top$, which implies that property (P_2) becomes

$$R\left(x, \frac{\partial S}{\partial x}, \frac{\partial H(x)}{\partial x}\right) = \gamma\left(x, \frac{\partial H(x)}{\partial x}\right) \frac{\partial S}{\partial x}^\top J_0 \frac{\partial H(x)}{\partial x}$$
$$= \gamma\left(x, \frac{\partial H(x)}{\partial x}\right) \frac{\partial H(x)}{\partial p} \ . \quad (8)$$

The function R is thus well-defined and can be derived from γ and H. To showcase the flexibility of the proposed PC-NODEs, we assume the Hamiltonian to be unknown and parametrize it as a single-layer NN with the form

$$H(x; \theta) = \log\left[\cosh(Kx + b)\right]^\top \mathbb{1}_n \ , \quad (9)$$

where $\mathbb{1}_n$ represents a column vector with n elements equal to 1, and $\theta = \{K, b\}$. Such an architecture is chosen for its elegance because it allows us to compute the required partial derivatives in closed form(Galimberti et al., 2021)

$$\frac{\partial H(x; \theta)}{\partial x} = K^\top \tanh(Kx + b) \ . \quad (10)$$

We parametrize γ as a single layer NN $\gamma : \mathbb{R}^8 \to \mathbb{R}^+$, where positivity is obtained by feeding the output through a sigmoid function, which is sufficient to ensure property (P_2). Finally, we assume the sparsity pattern of J_1 to be known, but not its parameters $\{\alpha, \beta\}$, to demonstrate how prior knowledge might be incorporated into the learning process [2].

Remark 4. Although PC-NODE (7) is slightly different from the generic representation in (1), the key system properties are still conserved. Indeed, one can always decompose the product between R and J in a sum of products without violating the first and second laws of thermodynamics as long as each term respects condition (2) and the skew-symmetry of J. See the proof of Proposition 1 for more details.

Remark 5. As we assume no thermal exchanges between the gas and the ambient air, according to the second law of thermodynamics, the gas entropy can never decrease.

4. APPLICATIONS AND RESULTS

This section presents the results obtained by fitting the two PC-NODEs described in the previous Section on real-world measurements for building thermal dynamics, and on simulated data for the gas piston system [3].

4.1 Building thermal dynamics

For the first application, we aim to identify the temperature dynamics of a residential apartment in NEST, a vertically integrated district in Duebendorf, Switzerland (Empa, 2021). It is composed of two bedrooms separated by a living room, as sketched in Fig. 2, where we neglected the impact of the two small bathrooms and processed the data similarly to Di Natale et al. (2022), including the computation of solar gains from horizontal irradiation measurements. Overall, three years of measurements of the temperature and solar gains in each zone, the respective heating and cooling powers, and the ambient temperature are available. The dataset has a sampling time of 15 min and was split into a training and a validation set containing time series of measurements truncated after three days, i.e. 288 steps, to alleviate the computational burden.

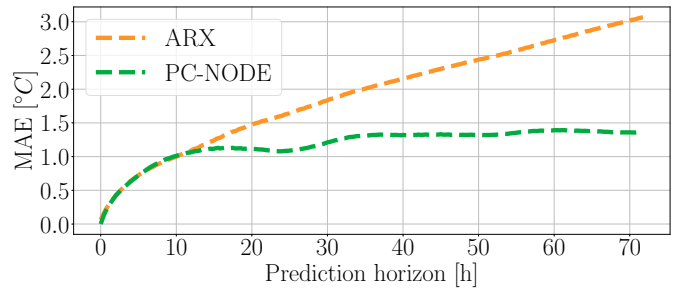

Fig. 4. MAE of the ARX model and PC-NODE over the prediction horizon averaged over the three zones and the validation time series.

To investigate the performance of the learned model, we analyze its accuracy over more than 750 sequences of three days of validation data. Averaged over the three zones and all the time series, the Mean Absolute Error (MAE) propagation over the 72 h horizon is depicted in Fig. 4, where the absolute error is computed as $||T_k - T(k)||_1$ for T_k the model prediction and $T(k)$ the measured temperature at each time step k. Since PC-NODE (6) is linear, we also plot the performance of a classical linear ARX model with 12 lag terms for reference, where the number of lags was tuned empirically and the parameters fitted to the data through least squares identification, similarly to Merema et al. (2022), for example.

As can be readily seen, thanks to the underlying physics captured by the Hamiltonian framework, the PC-NODE is able to fit the data significantly better, especially over long horizons. Indeed, it seems to be less prone to compounding errors: it improves the accuracy by 38.9% compared to the ARX on average over the entire prediction horizon, but this proportion rises to 55.8% at the end of the 72 h-long horizon.

In order to provide a visual comparison of the behavior of both models, we plot their temperature predictions over a sampled 72 h-long trajectory in August 2021 in Fig. 5. This figure hints that the ARX model is more sensitive to the various external gains, having a tendency to overestimate their impact. This can for example be observed towards the end of the horizon in Fig. 5, just before noon: when the sun rises, increasing the temperature of the building, the ARX cannot accurately capture this behavior, contrary to the PC-NODE. While only one sampled trajectory is presented in this paper for the sake of brevity, these effects generally hold across the validation dataset and explain the better performance of the PC-NODE.

[2] In the true system, α is the area of the piston and $\beta = 1$.

[3] The code and data can be found on `https://gitlab.nccr-automation.ch/loris.dinatale/pc-node`.

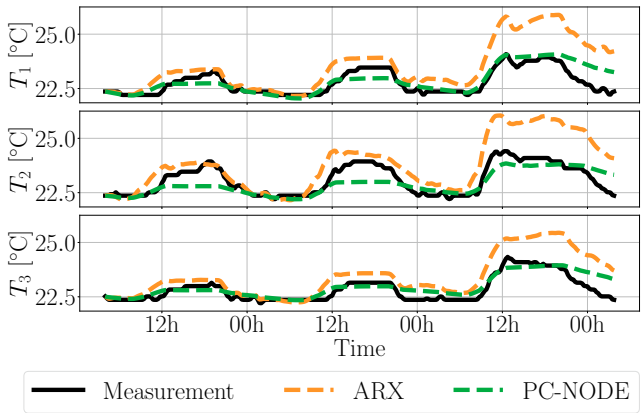

Fig. 5. August 18–20, 2021: Temperature predictions of the PC-NODE and ARX model on a sampled validation trajectory, compared to the true measurements.

4.2 Gas piston system

For the second application, we generated a synthetic dataset of 10′000 samples from system (7) using the `odeint` framework from `scipy`. The parameters are provided in Appendix C and the gas temperature has been computed as presented in Appendix A. To generate the training data, we then added Gaussian noise on each dimension d of the state, with $\epsilon_d \sim \mathcal{N}(0, 0.2\sigma_{x_d})$, where σ_{x_d} corresponds to the standard deviation of the dth dimension of x. Similarly to the previous application, the data was split into chunks of 250 steps to alleviate the computational burden. It was additionally normalized to ease the training of the NNs used in PC-NODE (7).

Despite not having access to the true Hamiltonian function and learning it as an NN from data, and even in the presence of white noise, PC-NODE (7) can accurately recover the position of the system, as pictured in Fig. 6 (bottom) for two sampled trajectories. However, a vanilla NODE, i.e. $\dot{x} = f_\theta(x)$ (Chen et al., 2018), where f_θ is a NN with two hidden layers of 32 neurons each, is also able to fit this data very well. On the other hand, the evolution of the entropy is more challenging to capture, as pictured on the top of Fig. 6, where we removed the noisy data for clarity. In that case, the PC-NODE clearly outperforms the vanilla NODE. Remarkably, the NODE sometimes predicts a decrease in entropy, which is inconsistent with the underlying physics (Remark 5) and does not happen with the PC-NODE.

5. CONCLUDING REMARKS

This work proposed PC-NODEs, NODEs endowed with IPH dynamics, to identify multi-physics systems from data. PC-NODEs are consistent with the first and second laws of thermodynamics by construction if simple conditions are respected, which allows one to rely on automatic BPTT to identify their parameters. Leveraging prior knowledge of these systems, the proposed framework demonstrated promising performance on both a thermal building modeling and a gas piston case study.

We believe these results can pave the way for large-scale distributed data-driven models with physical consistency guarantees.

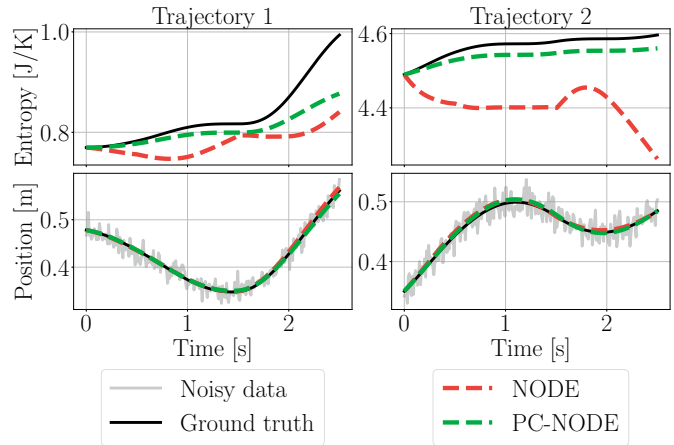

Fig. 6. Sampled trajectories of the piston position and the entropy of the gas ($\times 10^3$) over time.

REFERENCES

Chen, R.T., Rubanova, Y., Bettencourt, J., and Duvenaud, D. (2018). Neural ordinary differential equations. *arXiv preprint arXiv:1806.07366*.

Chen, Z., Zhang, J., Arjovsky, M., and Bottou, L. (2019). Symplectic recurrent neural networks. In *International Conference on Learning Representations*.

Cranmer, M., Greydanus, S., Hoyer, S., Battaglia, P., Spergel, D., and Ho, S. (2020). Lagrangian neural networks. In *ICLR 2020 Workshop on Integration of Deep Neural Models and Differential Equations*.

Di Natale, L., Svetozarevic, B., Heer, P., and Jones, C.N. (2022). Physically consistent neural networks for building thermal modeling: theory and analysis. *Applied Energy*, 325, 119806.

Empa (2021). NEST. Accessed: 01.10.2022. URL `https://www.empa.ch/web/nest/overview`.

Fazlyab, M., Morari, M., and Pappas, G.J. (2022). Safety verification and robustness analysis of neural networks via quadratic constraints and semidefinite programming. *IEEE Transactions on Automatic Control*, 67(1), 1–15.

Finzi, M., Wang, K.A., and Wilson, A.G. (2020). Simplifying Hamiltonian and Lagrangian neural networks via explicit constraints. *Advances in neural information processing systems*, 33, 13880–13889.

Galimberti, C.L., Furieri, L., Xu, L., and Ferrari-Trecate, G. (2021). Hamiltonian deep neural networks guaranteeing non-vanishing gradients by design. *arXiv preprint arXiv:2105.13205*.

Geirhos, R., Jacobsen, J.H., Michaelis, C., Zemel, R., Brendel, W., Bethge, M., and Wichmann, F.A. (2020). Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11), 665–673.

Greydanus, S., Dzamba, M., and Yosinski, J. (2019). Hamiltonian neural networks. *Advances in neural information processing systems*, 32.

Haber, E. and Ruthotto, L. (2017). Stable architectures for deep neural networks. *Inverse Problems*, 34(1), 014004.

Jin, P., Zhang, Z., Kevrekidis, I.G., and Karniadakis, G.E. (2022). Learning Poisson systems and trajectories of autonomous systems via Poisson neural networks. *IEEE Transactions on Neural Networks and Learning*

Systems.

Masi, F., Stefanou, I., Vannucci, P., and Maffi-Berthier, V. (2021). Thermodynamics-based artificial neural networks for constitutive modeling. *Journal of the Mechanics and Physics of Solids*, 147, 104277.

Merema, B., Saelens, D., and Breesch, H. (2022). Demonstration of an MPC framework for all-air systems in nonresidential buildings. *Building and Environment*, 217, 109053.

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., and Lerer, A. (2017). Automatic differentiation in pytorch. *NIPS 2017 Autodiff Workshop*.

Ramirez, H., Maschke, B., and Sbarbaro, D. (2013a). Irreversible port-Hamiltonian systems: A general formulation of irreversible processes with application to the CSTR. *Chemical Engineering Science*, 89, 223–234.

Ramirez, H., Maschke, B., and Sbarbaro, D. (2013b). Modelling and control of multi-energy systems: An irreversible port-Hamiltonian approach. *European journal of control*, 19(6), 513–520.

Rubanova, Y., Chen, R.T., and Duvenaud, D.K. (2019). Latent ordinary differential equations for irregularly-sampled time series. *Advances in neural information processing systems*, 32.

Van der Schaft, A. and Jeltsema, D. (2021). On Energy Conversion in Port-Hamiltonian Systems. *arXiv preprint arXiv:2103.09116*.

Wang, R. and Yu, R. (2021). Physics-guided deep learning for dynamical systems: A survey. *arXiv preprint arXiv:2107.01272*.

Werbos, P.J. (1990). Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10), 1550–1560.

Zakwan, M., Xu, L., and Ferrari Trecate, G. (2022). Robust classification using contractive Hamiltonian neural ODEs. *IEEE Control Systems Letters*, 7, 145–150.

Appendix A. TEMPERATURE COMPUTATION

By definition, the energy of a mass m of air can be described as $E(T) = mc(T)T$, where c is the specific heat capacity of air and T is temperature. Assuming we deal with an ideal gas, we also know that $PV = nRT$. This allows us to rewrite the time derivative of entropy, by definition satisfying

$$\begin{aligned}\frac{dS}{dt} &= \frac{1}{T}\frac{dE}{dt} + \frac{P}{T}\frac{dV}{dt} \\ &= \frac{1}{T}\frac{d}{dt}(mc(T)T) + \frac{nRT}{VT}\frac{d}{dt}V \,.\end{aligned}$$

Since the temperature does not change abruptly, we can assume a constant heat capacity $c(T) \approx c$ and obtain

$$\frac{dS}{dt} \approx mc\frac{\frac{dT}{dt}}{T} + nR\frac{\frac{dV}{dt}}{V} = mc\frac{d\ln T}{dt} + nR\frac{d\ln V}{dt} \,. \quad \text{(A.1)}$$

Integrating (A.1) on both sides from an initial time t_i to a final time t_f, we get

$$\int_{t_i}^{t_f} \frac{dS}{dt}dt = mc\int_{t_i}^{t_f} \frac{d\ln T}{dt}dt + nR\int_{t_i}^{t_f} \frac{d\ln V}{dt}dt,$$

leading to

$$\begin{aligned}S(t_f) - S(t_i) &= mc\ln\frac{T(t_f)}{T(t_i)} + nR\ln\frac{V(t_f)}{V(t_i)} \\ &= \ln\left[\left(\frac{T(t_f)}{T(t_i)}\right)^{mc}\left(\frac{V(t_f)}{V(t_i)}\right)^{nR}\right] .\end{aligned}$$

We can thus compute the final temperatures as

$$T(t_f) = \left[\exp\left(\frac{S(t_f) - S(t_i)}{mc}\right)\left(\frac{V(t_f)}{V(t_i)}\right)^{\frac{-nR}{mc}}\right]T(t_i) \,.$$

Appendix B. PROOF OF PROPOSITION 1

Let us define $\mathcal{E} = \{(i,j)|$ Zones i and j are adjacent$\}$, the set of connections between the thermal zones, and consider the following decomposition of $\tilde{J}(T)$:

$$\tilde{J}(T) = \sum_{k\in\mathcal{E}} R_k(T)\mathcal{J}_k \,,$$

where $R_k : \mathbb{R}^N \mapsto \mathbb{R}$, $R_k(T) = \lambda_{ij}\frac{(T_j - T_i)}{(T_iT_j)}$ and $\mathcal{J}_k$ is an $N \times N$ constant skew-symmetric matrix with zeroes everywhere, except $(J_k)_{ij} = -(J_k)_{ji} = 1$, for $k = (i,j)$.

Then, for $T_e, Q_s, Q_h, Q_c \equiv 0$, we have:

$$\begin{aligned}\frac{dH}{dt} &= \frac{\partial H(S)}{\partial S}^\top \dot{S} = \frac{\partial H(S)}{\partial S}^\top \left[\sum_{k\in\mathcal{E}} R_k(T)\mathcal{J}_k\right]\frac{\partial H(S)}{\partial S} \\ &= \sum_{k\in\mathcal{E}}\left[\frac{\partial H(S)}{\partial S}^\top R_k(T)\mathcal{J}_k\frac{\partial H(S)}{\partial S}\right] = 0 \,,\end{aligned}$$

since each term of the sum is zero, as in equation (3), because each $\mathcal{J}_k$ is now constant and each $R_k(T)$ satisfies condition (P_2). This proves the required conservation of energy of the system.

In order to verify the irreversible creation of total entropy S_t of the system, we note that for $T_e, Q_s, Q_h, Q_c \equiv 0$:

$$R_k(T) = \left(\frac{\lambda_{ij}}{T_iT_j}\right)\{S_t, H\}_{\mathcal{J}_k} \,,$$

as in the case of two heat exchangers (Ramirez et al., 2013a). Since the total entropy is the sum of the entropy in each zone d, we get

$$\begin{aligned}\dot{S}_t &= \sum_{d=1}^{n}(\dot{S})_d = \sum_{d=1}^{n}\left(\left[\sum_{k\in\mathcal{E}} R_k\mathcal{J}_k\right]\frac{\partial H(S)}{\partial S}\right)_d \\ &= \sum_{k\in\mathcal{E}} R_k\sum_{d=1}^{n}\left(\mathcal{J}_k\frac{\partial H}{\partial S}\right)_d = \sum_{k\in\mathcal{E}} R_k\left(\mathbb{1}_n^\top\mathcal{J}_k\frac{\partial H}{\partial S}\right) \\ &= \sum_{k\in\mathcal{E}} R_k\left(\frac{\partial S_t}{\partial S}^\top\mathcal{J}_k\frac{\partial H}{\partial S}\right) = \sum_{k\in\mathcal{E}}\frac{\lambda_{ij}}{T_iT_j}\{S_t, H\}^2_{\mathcal{J}_k} \geq 0 \,,\end{aligned}$$

since $\frac{\partial S_t}{\partial S} = \mathbb{1}_n$ by definition, and the inequality holds if all $\{\lambda_{ij}\}_{(i,j)\in\mathcal{E}}$ are positive since temperatures are positive.

Finally, if all the input matrices B_e, B_s, B_h, and B_c are positive definite, monotonicity follows from the fact that PC-NODE (6) is affine in input by construction. □

Appendix C. GAS PISTON PARAMETERS

The gas piston system was simulated from $T(0) = 290\,\mathrm{K}$, $x(0) = [0, 0.001, 0.3, 0]^\top$, with $m = 5\,\mathrm{kg}$, area $\alpha = 0.033\,\mathrm{m}^2$, $\beta = 1$, $\mu = 1$, and $K = 10\,\mathrm{N\,m^{-1}}$, and the sampling time $h = 0.01\,\mathrm{s}$.