MITSUBISHI ELECTRIC RESEARCH LABORATORIES
https://www.merl.com

Learning Residual Dynamics via Physics-Augmented Neural Networks: Application to Vapor Compression Cycles

Chinchilla, Raphael; Deshpande, Vedang M.; Chakrabarty, Ankush; Laughman, Christopher R.

TR2023-051 May 31, 2023

Abstract

In order to improve the control performance of vapor compression cycles (VCCs), it is often necessary to construct accurate dynamical models of the underlying thermo-fluid dynamics. These dynamics are represented by complex mathematical models that are composed of large systems of nonlinear and numerically stiff differential algebraic equations (DAEs). The effects of nonlinearity and stiffness may be ameliorated by using physics-based models to describe characteristic system behaviors, and approximating the residual (unmodeled) dynamics using neural networks. In these so-called 'physics-augmented' or 'physics-informed' machine learning approaches, the learning problem is often solved by jointly estimating parameters of the physics component model and weights of the network. Furthermore, such approaches also often assume the availability of full-state information, which typically are not available in practice for energy systems such as VCCs after deployment. Rather than concurrently performing state/parameter estimation and network training, which often leads to numerical instabilities, we propose a framework for decoupling the network training from the joint state/parameter estimation problem by employing state-constrained Kalman smoothers customized for VCC applications. We show the effectiveness of our proposed framework on a Julia-based, high-fidelity simulation environment calibrated to a model of a commercially-available VCC and achieve an accuracy of 98% calculated over 24 states and multiple initial conditions under realistic operating conditions.

American Control Conference (ACC) 2023

Mitsubishi Electric Research Laboratories, Inc.
201 Broadway, Cambridge, Massachusetts 02139

Learning Residual Dynamics via Physics-Augmented Neural Networks: Application to Vapor Compression Cycles

Raphael Chinchilla[1], Vedang M. Deshpande[2,†], Ankush Chakrabarty[2], and Christopher R. Laughman[2]

***Abstract*— In order to improve the control performance of vapor compression cycles (VCCs), it is often necessary to construct accurate dynamical models of the underlying thermo-fluid dynamics. These dynamics are represented by complex mathematical models that are composed of large systems of nonlinear and numerically stiff differential algebraic equations (DAEs). The effects of nonlinearity and stiffness may be ameliorated by using physics-based models to describe characteristic system behaviors, and approximating the residual (unmodeled) dynamics using neural networks. In these so-called 'physics-augmented' or 'physics-informed' machine learning approaches, the learning problem is often solved by jointly estimating parameters of the physics component model and weights of the network. Furthermore, such approaches also often assume the availability of full-state information, which typically are not available in practice for energy systems such as VCCs after deployment. Rather than concurrently performing state/parameter estimation and network training, which often leads to numerical instabilities, we propose a framework for decoupling the network training from the joint state/parameter estimation problem by employing state-constrained Kalman smoothers customized for VCC applications. We show the effectiveness of our proposed framework on a Julia-based, high-fidelity simulation environment calibrated to a model of a commercially-available VCC and achieve an accuracy of 98% calculated over 24 states and multiple initial conditions under realistic operating conditions.**

***Index Terms*— Energy systems, physics-informed machine learning, state estimation, data-driven methods, system identification, HVAC.**

I. INTRODUCTION

Vapor compression cycles are prevalent in a diverse array of applications due to their ability to provide heating and cooling capacity over a wide range of temperatures. They are positioned to play an increasingly important role in our society, as they offer an effective means to advance decarbonization goals by replacing fossil fuel-based heating systems and utilizing the power generated by renewable energy resources. Unfortunately, vapor compression cycles are also responsible for 10% of total annual electricity consumption in US, which is attributed to space cooling in buildings alone, and many common refrigerants used in such equipment have global warming potential hundreds to thousands of times higher than CO_2 [1]. Because of their wide-spread use, technologies to improve the performance of vapor compression cycles have the potential to make a significant impact in the future.

A range of control technologies employed in vapor compression cycles use computational models that represent the underlying physics of the system for prediction, e.g., model predictive control [2] and estimation of performance parameters such as cooling capacity delivered by the heat exchangers (HEXs) [3]. Models of vapor compression cycles also play an important role in the development of fault detection and diagnosis algorithms, and are central to emerging digital twin technologies [4] aiming to improve energy efficiency, streamline maintenance, and maximize the occupant comfort.

Longstanding efforts have been focused on developing physics-based descriptions of vapor-compression cycle dynamics, at both the component and system level, for the purposes of obtaining good predictive models for this variety of applications [5], [6]. These models are generally formulated to satisfy application-specific physics-based or computational requirements and relate to the physical processes most relevant to the application. The simplifying assumptions that accompany these requirements, such as lumped parameters and finite discretizations, often lead to a mismatch between the model and data collected from the underlying physical system. Addressing this mismatch to improve the accuracy of model predictions is often a significant focus in model development processes.

Recent advances in machine learning methods and improvements in computational resources have continued to motivate work in data-driven modeling and identification of physical systems. Pure data-driven treatments in which the underlying system is modeled using a pre-specified dictionary of basis functions [7], basis functions partially derived based on the data [8], or a neural network as a black box [9], are popular approaches because of their system-agnostic perspective. Unfortunately, these methods are rarely applicable to vapor-compression systems because these methods often need large datasets containing full-state trajectories to achieve reasonable modeling accuracy that are unavailable for field-installed equipment due to limited sensor data. Moreover, purely data-driven methods often suffer from a combination of other challenges such as non-interpretability of the model, large parameter search spaces, and do not explicitly account for the fundamental physical laws that govern the system.

Hybrid modeling paradigms that combine physics-based models with data-driven components, e.g., neural networks, to learn the behavior of a dynamical system and thereby circumvent some of the shortcomings of purely data-driven

[1]Center for Control, Dynamical Systems, and Computation, at University of California, Santa Barbara. This work was completed during a research internship at Mitsubishi Electric Research Laboratories. raphaelchinchilla@ucsb.edu

[2]Mitsubishi Electric Research Laboratories. {deshpande,chakrabarty,laughman}@merl.com

[†]Corresponding author. +1 (617) 621-7548.

approaches have been recently gaining in popularity [10]–[16]. One focus area has been in employing a neural network as an additive term to a physics based model for function approximation. In particular, the APHINITY framework presented in [10] is designed such that most of the information comes from the physics-based model while minimizing contributions of the neural network. Because this method requires noiseless observations of all the states in the system, the authors of [12] modify the APHYNITY method by augmenting the cost function with a penalization between the output of the learned system and the real observations. Similarly, [11] modifies APHINITY to use partial observations in learning by introducing bounds on the contribution of the neural network and statistical model to control the contribution of each one.

The APHYNITY method and its variations concurrently learn the optimal value of the physics-based model parameters and neural network weights by solving a constrained optimization problem. Such simultaneous physical-parameter estimation and neural network training can be challenging for vapor-compression cycles because the training process often leads to instabilities within numerically stiff models. Concurrent methods are also difficult to apply to many VCC models, as they are typically implemented in a modeling and simulation environment that is different from the machine learning ecosystem. In the this paper, we address some of these challenges associated with the existing approaches.

The **contributions** of this work are twofold. In Section II, we first present a framework in which the problems of physical-model parameter estimation and neural network training are decoupled into two optimization problems, which are easier to solve separately and can accommodate partial and noisy state observations. In this formulation, the first optimization problem essentially becomes a joint state and parameter estimation problem that can be solved using nonlinear state estimation tools such as state-constrained Kalman smoothers. In the second optimization problem, the neural network performs a regression to learn the residual dynamics, enabling the use of machine learning methods that may not be applied directly to the physics-based model. These two optimization problems can be potentially solved in different simulation environments that allow exchange of data. While the proposed formulations are motivated by applications in VCC, they have wide applicability to a multitude of similar problems. Secondly, the proposed approach is applied to a model of a commercially-available VCC in Section III. While significant work has been done on physics-based learning for energy and buildings systems [17]–[19], only a limited set of explorations of neural networks to vapor-compression cycles have been investigated, such as for modeling component-level dynamics of HEXs [20], and certain cycle parameters such as steady-state performance coefficients [21] and mass flow rates [22]. System-level hybrid modeling of cycles to predict internal states using neural networks has been largely an unexplored topic, which we address in the present study. Concluding remarks are provided in Section IV.

II. Learning Residual Dynamics

We consider an abstracted dynamical model

$$x^+ = F(x, u) \tag{1}$$

where $F : \mathcal{X} \times \mathcal{U} \to \mathcal{X}$ is an unknown map, with $\mathcal{X} \subset \mathbb{R}^{n_x}$ being the domain of state trajectories and $\mathcal{U} \subset \mathbb{R}^{n_u}$ being the domain of admissible controls. This definition can be extended to accommodate an explicit dependence on time variable, if required. Based on our physical understanding of thermo-fluid dynamics, we can model parametric representations of F as $F_p : \mathcal{X} \times \mathcal{U} \times \Theta \to \mathcal{X}$, where Θ is the space of physics-based model parameters. While these representations tend to be somewhat accurate and interpretable, there remain unmodeled dynamics $F - F_p$, referred to herein as a *residual model*.

Improving the accuracy of F_p to reduce the magnitude of the residual dynamics is often expensive and incurs significant modeling complexity or extensive experimentation on the VCC, both of which are often impractical. Alternatively, one could use a function approximator to learn the residual dynamics. While the theoretical results presented in this section do not make assumptions on the type of function approximator, the results are developed with neural networks as potential approximators because of their ability to approximate arbitrarily complex functions.

More precisely, let $F_{nn} : \mathcal{X} \times \mathcal{U} \times \mathcal{W} \to \mathcal{X}$ be a neural network with weights lying in the space $\mathcal{W}$. Once the neural architecture for F_{nn} is set, an accurate physics-augmented[1] neural network can be built by computing parameters θ^* and w^* such that

$$F(x, u) = F_p(x, u, \theta^*) + F_{nn}(x, u, w^*), \tag{2}$$

for all $(x, u) \in \mathcal{X} \times \mathcal{U}$. Clearly, one cannot evaluate the VCC for all possible combinations of $\mathcal{X} \times \mathcal{U}$, therefore, it is more practical to choose an architecture of F_{nn} and then find optimal parameters θ^*, w^* for a finite data set and expect the model to generalize over all $\mathcal{X} \times \mathcal{U}$.

In general, it is not possible to measure all the states of VCCs, as one is often restricted to the availability of specific measurements. Moreover, commercially-available sensors do not have infinite precision, which is mathematically translated as an additive noise to the measurement. Finally, system are often subject to external site-specific factors that are not included in a general model of a vapor compression cycle. These may include the effects of particular pipe lengths or configurations, the effects of local airflow patterns, or installation-specific considerations that result from the construction of the system at a particular geographic location. These effects are not modeled using domain knowledge, and are instead interpreted as random process disturbances.

In the following subsection, we assume the availability of full state information with no noise or disturbances to begin our exposition. In subsequent subsections, we consider

[1]We refrain from referring to this as a 'physics-informed' neural network (PINN) because the PINN terminology admits a specific network and training paradigm; c.f. [23].

the case of noisy partial state observations, and conclude by presenting a method for learning unmodeled dynamics in the case where the system is subject to noisy partial observations and process disturbances.

Remark 1 (Discrete-time vs. continuous-time): While we consider F_p to be a discrete-time system, it is generally obtained from the solution of an ordinary differential equation, that is $F_p(x,u,\theta) = x + \int_t^{t+\Delta_t} f_p(x(\tau),u,\theta)\,\mathrm{d}\tau$, where $f_p(x,u,\theta)$ is a continuous-time model. Most formulations that combine neural networks with physics-based models directly use a continuous-time model, which has the advantage that the neural network is not trained for a specific sampling time. However, as is the case with VCCs, many industrial process models $f_p(\cdot)$ are numerically stiff and tend to be very sensitive to numerical perturbations in the dynamics. They are often built using a combination of analytical equations, splines, and lookup tables and have to operate strictly in a given regions in state space. Once the neural network is included, the solution of the differential equation can violate the operational conditions, especially during training, leading to simulation failures. ◁

A. Noise-free Full State Observations

Suppose we are able to obtain noise-free full state observation of the solution of (1). We will denote by $x_k^{(n)}$ the observed solution at time t_k given the initial condition $x^{(n)}(0)$ of n^{th} trajectory in the data set. Inspired by the APHYNITY framework for continuous-time proposed in [10], one can obtain parameters θ^* and network weights w^* by solving

$$\begin{aligned} \min_{\theta\in\Theta, w\in\mathcal{W}} & \sum_{n=1}^{N}\sum_{k=1}^{K} \left\| F_{nn}(x_k^{(n)}, u_k^{(n)}, w)\right\|_{P^{-1}}^2 \\ \text{s.t.} \quad & x_{k+1}^{(n)} = F_p(x_k^{(n)}, u_k^{(n)}, \theta) + F_{nn}(x_k^{(n)}, u_k^{(n)}, w) \end{aligned} \tag{3}$$

for all $k = 1,\cdots,K$ and $n = 1,\cdots,N$. Here, $P \succ 0$ is a weight matrix, $\|v\|_P^2 \triangleq v^\top P v$, and $F_{nn}(\cdot)$ is assumed to be expressive enough to satisfy the equality constraints. The practical implications regarding the satisfaction of this constraint are discussed at the end of this subsection.

The rationale for using (3) instead of simply minimizing $\left\| x_{k+1}^{(n)} - F_p(x_k^{(n)}, u_k^{(n)}, \theta) - F_{nn}(x_k^{(n)}, u_k^{(n)}, w)\right\|^2$ is that this cost can lead to undesired solutions. For instance, a combination of parameters (θ^*, w^*) can exist such that $F_p(x_k^{(n)}, u_k^{(n)}, \theta^*) = 0$ and all the dynamics information comes from the neural network. Instead, we desire that the physics-based model $F_p(\cdot)$ provide most of the explanation of the system dynamics because it tends to generalize better to unseen data. In the limiting case, if there is a parameter θ^* such that $F(x,u) = F_p(x,u,\theta^*)\ \forall (x,u) \in \mathcal{X}\times\mathcal{U}$, then (3) will find such parameters and discard any contribution from the network.

The numerical optimization required to solve (3) tends to be challenging. First, the equality constraints either have to be relaxed or managed via Lagrangian methods, as proposed in [10]. Second, θ and w are fundamentally different optimization variables. On one hand, θ in general has low dimension but derivatives that are hard to compute, as they require the derivative of the solution of a differential equation. Moreover, the dynamics of the vapor-compression cycle are nonlinear, numerically stiff, and have derivative discontinuities due to the phase changes that accompany evaporation or condensation. On the other hand, w has large dimension, but the derivatives can be efficiently computed using standard computing tools such as automatic differentiation.

These challenges can be addressed by the result from the following proposition, which allows us to decouple (3) into two tractable optimization problems.

Proposition 1 (Equivalent two optimizations): Let $\Omega \triangleq \bigcup\{(\theta^*, w^*)\}$, where pair (θ^*, w^*) denotes any minimizer of (3). Suppose $\theta^\star$ be a solution to the problem

$$\min_{\theta\in\Theta} \sum_{n=1}^{N}\sum_{k=1}^{K} \left\| x_{k+1}^{(n)} - F_p(x_k^{(n)}, u_k^{(n)}, \theta)\right\|_{P^{-1}}^2, \tag{4a}$$

and $w^\star$ be a solution to the optimization problem

$$\min_{w\in\mathcal{W}} \sum_{n=1}^{N}\sum_{k=1}^{K} \left\| \delta_k^{(n)*} - F_{nn}(x_k^{(n)}, u_k^{(n)}, w)\right\|^2, \tag{4b}$$

where the residual $\delta_k^{(n)*} \triangleq x_{k+1}^{(n)} - F_p(x_k^{(n)}, u_k^{(n)}, \theta^*)$, then $(\theta^\star, w^\star) \in \Omega$. ◁

Proof: For any $(\theta^*, w^*) \in \Omega$, if one takes $F_{nn}(x_k^{(n)}, u_k^{(n)}, w^*) = \delta_k^{(n)*}$ then $(\theta^*, \delta_k^{(n)*})$ is a solution of

$$\begin{aligned} \min_{\theta\in\Theta, \delta_k^{(n)}\in\mathcal{D}_k^{(n)}} & \sum_{n=1}^{N}\sum_{k=1}^{K} \left\| \delta_k^{(n)}\right\|_{P^{-1}}^2 \\ \text{s.t.: } & x_{k+1}^{(n)} = F_p(x_k^{(n)}, u_k^{(n)}, \theta) + \delta_k^{(n)}, \end{aligned} \tag{5}$$

for all $k = 1,\ldots,K$, and $n = 1,\ldots,N$ where $\mathcal{D}_k^{(n)} := \{F_{nn}(x_k^{(n)}, u_k^{(n)}, w)\ \forall w \in \mathcal{W}\}$. To obtain (4a) from (5), we solve for $\delta_k^{(n)}$ in the equality constraints and substitute back into the cost function.

After solving (4a), to obtain w^*, we substitute $x_{k+1}^{(n)} - F_p(x_k^{(n)}, u_k^{(n)}, \theta)$ by $\delta_k^{(n)*}$ in (3):

$$\begin{aligned} \min_{w\in\mathcal{W}} & \sum_{n=1}^{N}\sum_{k=1}^{K} \left\| F_{nn}(x_k^{(n)}, u_k^{(n)}, w)\right\|_{P^{-1}}^2 \\ \text{s.t.: } & F_{nn}(x_k^{(n)}, u_k^{(n)}, w) = \delta_k^{(n)} \\ & \forall k = 1,\ldots,K, \text{ and } n = 1,\ldots,N. \end{aligned} \tag{6}$$

By definition, we can rewrite this expression as

$$\begin{aligned} \min_{w\in\mathcal{W}} & \sum_{n=1}^{N}\sum_{k=1}^{K} \left\| \delta_k^{(n)}\right\|_{P^{-1}}^2 \\ \text{s.t.: } & F_{nn}(x_k^{(n)}, u_k^{(n)}, w) = \delta_k^{(n)} \\ & \forall k = 1,\ldots,K, \text{ and } n = 1,\ldots,N \end{aligned} \tag{7}$$

As $\delta_k^{(n)}$ is a constant, this optimization is just equivalent to find w such that the equality constraint is satisfied, which is the same as solving (4b). ■

A crucial point in the proof is the (implicit) assumption that the neural network is expressive enough such that the non-linear equation in w, namely $F_{nn}(x_k^{(n)}, u_k^{(n)}, w) = \delta_k^{(n)}$ has a solution. This assumption is already present in the formulation of (3) by assuming the equality constraint is feasible. Since neural networks are universal function approximators, in principle, there exist neural architectures and corresponding weights that will satisfy such feasibility conditions. However, it is practically impossible to determine such a neural network with finite data. As we mentioned before, in APHYNITY [10] this potential feasibility issue is addressed by effectively relaxing the equality constraint when numerically solving the problem. In our approach, the possible limited expressiveness of F_{nn} can be addressed by accepting solutions of (4b) achieving a cost smaller than the specified tolerance, which typically requires a search over different F_{nn} architectures and space of hyperparameters.

B. Noisy Partial State Observations

In the previous subsection, we assume that perfect observations of all the states of the system are available. However, only partial and noisy observations of the states are available in VCCs and many other applications, which are represented by the observation equation

$$y = h(x) + \nu \tag{8}$$

where ν is a zero-mean white Gaussian process with covariance R and $h(\cdot)$ is a state observation function. Moreover, the initial state is generally not known, so one assumes a prior distribution on the initial state x_0 given by a Gaussian distribution with mean $\bar{x}_0$ and covariance S. Finally, even if the states are not observed, physics-based considerations often stipulate that they belong to a set of admissible states $\mathcal{X}$, such as the positive cone. Let $y_k^{(n)}$ be observations of the real states $x_k^{(n)}$ according to (8) given initial condition $x_0^{(n)}$. Inspired by (3), we can construct the optimization problem

$$\begin{aligned}
&\min_{\theta\in\Theta, w\in\mathcal{W}, \hat{x}_k^{(n)}\in\mathcal{X}} \sum_{n=1}^{N}\sum_{k=0}^{K} \left\| F_{nn}(\hat{x}_k^{(n)}, u_k^{(n)}, w)\right\|^2_{P^{-1}} \\
&+ \left\| y_k^{(n)} - h(\hat{x}_k^{(n)})\right\|^2_{R^{-1}} + \left\| \bar{x}_0 - \hat{x}_0^{(n)}\right\|_{S^{-1}} \\
&\text{s.t.: } \hat{x}_{k+1}^{(n)} = F_p(\hat{x}_k^{(n)}, u_k^{(n)}, \theta) + F_{nn}(\hat{x}_k^{(n)}, u_k^{(n)}, w) \\
&\forall k = 0, \dots, K-1, \text{ and } n = 1, \dots, N
\end{aligned} \tag{9}$$

This expression is a generalization of (3) in the following sense: Supposing that $h(x) = x$, there is no observation noise, and the initial condition $x_0^{(n)}$ is known and equal to $\bar{x}_0$, then $R = 0I$ and $S = 0I$ where I is the identity matrix. If we use the convention that $(0I)^{-1} = +\infty I$, this is equivalent to enforcing the equality constraint $h(\hat{x}_k^{(n)}) = y_k^{(n)} \implies \hat{x}_k^{(n)} = x_k^{(n)}$, which, if we substitute back in (9), gives (3).

Proposition 2: Let $\Omega' \triangleq \bigcup\{(\theta^*, w^*, \hat{x}_k^{(n)*})\}$, where the tuple $(\theta^*, w^*, \hat{x}_k^{(n)*})$ denotes any minimizer of (9). Supposing that $(\theta^*, \hat{x}_k^{(n)*})$ is a solution to the problem

$$\begin{aligned}
&\min_{\theta\in\Theta, \hat{x}_k^{(n)}\in\mathcal{X}} \sum_{n=1}^{N}\sum_{k=0}^{K-1} \left\| \hat{x}_{k+1}^{(n)} - F_p(\hat{x}_k^{(n)}, u_k^{(n)}, \theta)\right\|^2_{P^{-1}} \\
&+ \left\| y_k^{(n)} - h(\hat{x}_k^{(n)})\right\|^2_{R^{-1}} + \left\| \bar{x}_0 - \hat{x}_0^{(n)}\right\|_{S^{-1}}
\end{aligned} \tag{10a}$$

and $w^\star$ be a solution to the optimization problem

$$\min_{w\in\mathcal{W}} \sum_{n=1}^{N}\sum_{k=1}^{K} \left\| \delta_k^{(n)*} - F_{nn}(\hat{x}_k^{(n)^*}, u_k^{(n)}, w)\right\|^2 \tag{10b}$$

where the residual $\delta_k^{(n)*} = \hat{x}_{k+1}^{(n)*} - F_p(\hat{x}_k^{(n)*}, u_k^{(n)}, \theta^*)$, then $(\theta^\star, w^\star, x_k^{(n)*}) \in \Omega'$. ◁

Proof: The proof is similar to that of Proposition 1 and is briefly outlined below.

First, $F_{nn}(\hat{x}_k^{(n)}, u_k^{(n)}, w)$ in the objective of (9) is eliminated using the equality constraints, which gives us (10a). Then the residuals calculated using $\delta_k^{(n)*} = \hat{x}_{k+1}^{(n)*} - F_p(\hat{x}_k^{(n)*}, u_k^{(n)}, \theta^*)$ must satisfy the equality constraints in (9), which gives us an equivalent feasibility problem (10b). ■

Remark 2: Equation (10a) can be interpreted as a maximum a posteriori estimation of θ and x, where the dynamical system is subject to Gaussian disturbances (with a uniform (possibly improper) prior on the domain of θ). Instead of explicitly solving (10a) one can obtain approximate θ^* and $\hat{x}^{(n)*}$ using optimal smoothing and data assimilation methods, such as Kalman smoothers or 4DVar [24], which can be customized for specific VCC applications [3] for substantially faster solutions. ◁

C. Noisy Partial State Observations with Process Disturbances

To account for the stochastic disturbances in the real system, the model (1) is updated to add a disturbance term, i.e.

$$x^+ = F(x, u) + d \tag{11}$$

where d zero-mean white Gaussian process with covariance Q. If we applied (9) or (10) to learn the missing dynamics from data from such system, the neural network $F_{nn}(\cdot)$ would try to over fit and learn these disturbances. This can be addressed elegantly when using an approach which formulates two optimization problems.

First, one can obtain an optimal $(\theta^*, \hat{x}_k^{(n)*})$ by solving

$$\begin{aligned}
&\arg\min_{\theta\in\Theta, \hat{x}_k^{(n)}\in\mathcal{X}} \sum_{n=1}^{N}\sum_{k=0}^{K-1} \left\| \hat{x}_{k+1}^{(n)} - F_p(\hat{x}_k^{(n)}, u_k^{(n)}, \theta)\right\|^2_{P^{-1}} \\
&+ \left\| y_k^{(n)} - h(\hat{x}_k^{(n)})\right\|^2_{R^{-1}} + \left\| \bar{x}_0 - \hat{x}_0^{(n)}\right\|_{S^{-1}}
\end{aligned} \tag{12a}$$

which is the same optimization as (10a). It does not need to be modified as it already implicitly assumes that $F_p(\cdot)$ cannot fully describe the dynamics.

Second, to avoid over fitting the neural network that could happen if using (4b) or (10b), we modify the neural network

learning problem to

$$\min_{w} \sum_{n=1}^{N} \sum_{k=1}^{K} \left\| F_{nn}(\hat{x}_k^{(n)*}, u_k^{(n)}, w^*) \right\|_{P^{-1}}^2 + \left\| F_{nn}(\hat{x}_k^{(n)*}, u_k^{(n)}, w^*) - \delta_k^{(n)*} \right\|_{Q^{-1}}^2 \quad \text{(12b)}$$

$$\text{s.t.: } F_p(\hat{x}_k^{(n)*}, u_k^{(n)}, \theta^*) + F_{nn}(\hat{x}_k^{(n)*}, u_k^{(n)}, w) \in \mathcal{X}.$$

This second expression is motivated by (6) in the proof of Proposition 1. To obtain (12b), we relax the equality constraint $F_{nn}(x_k^{(n)*}, u_k^{(n)}, w^*) = \delta_k^{(n)*}$, which is the cause for the over fitting, by including it in the cost with the weight Q^{-1}. Notice that, similar to the case of partial and noisy state observations, we can see this expression as a generalization of the case with no process disturbance. No process disturbance is equivalent to $Q = 0I$, so if we use the convention $Q^{-1} = \infty I$ we retrieve the expression with equality constraints, where there is an infinite cost if the constraint is not respected.

While (12b) has the constraint $F_p(\hat{x}_k^{(n)*}, u_k^{(n)}, \theta^*) + F_{nn}(\hat{x}_k^{(n)*}, u_k^{(n)}, w) \in \mathcal{X}$, we will see in the next subsection that satisfaction of this constraint is not fundamental. If $\mathcal{X} := \{x : \mathbb{R}^n : G(x) = 0, F(x) \leq 0\}$, a simple alternative is to include the penalization $\mu\|G(x)\|^2 + \mu\|\max(0, F(x))\|^2$ with μ sufficiently large such that the constraint is approximately satisfied.

Remark 3: In general, the optimizers in all three scenarios above (*i.e.,* full state observation, partial state observation and partial state observation with process disturbances) do not guarantee that $F_p(x, u, \theta^*) + F_{nn}(x, u, w^*) \in \mathcal{X}$ for an arbitrary feasible pair (x, u), especially if it is outside of the training data set. This can be addressed by passing the learned model through a projection operator, *i.e.,*

$$x^+ = \Pi_{\mathcal{X}}\Big(F_p(x, u, \theta) + F_{nn}(x, u, w)\Big), \quad \text{(13)}$$

where $\Pi_{\mathcal{X}}(\cdot)$ is projection operator to the set $\mathcal{X}$ defined as

$$\Pi_{\mathcal{X}}(x) \in \arg\min_{\bar{x} \in \mathcal{X}} \|x - \bar{x}\|.$$

While in theory this requires rewriting all of the optimizations in this section using (13) as the dynamical model, this is usually not necessary because most of the trajectory points $x_k^{(n)}$ in the training data will be in the interior of $\mathcal{X}$. ◁

We demonstrate the results of the approach presented in this section using a simple illustrative example in the next subsection, in which we consider a pendulum with damping which is a common example in the literature of learning unmodeled dynamics. The case study on VCC will be discussed in Section III.

D. Illustrative Example: Pendulum with Damping

We consider a pendulum system complemented with process disturbances and noisy observations described to obtain

$$\begin{bmatrix}\phi^+ \\ \omega^+\end{bmatrix} = \begin{bmatrix}\phi \\ \omega\end{bmatrix} + \int_t^{t+\Delta_t} \begin{bmatrix}\omega(\tau) \\ -\psi\phi(\tau) - \alpha\omega(\tau)\end{bmatrix} d\tau + d \quad \text{(14a)}$$

$$y = \begin{bmatrix}\phi & \omega\end{bmatrix}^T + \nu \quad \text{(14b)}$$

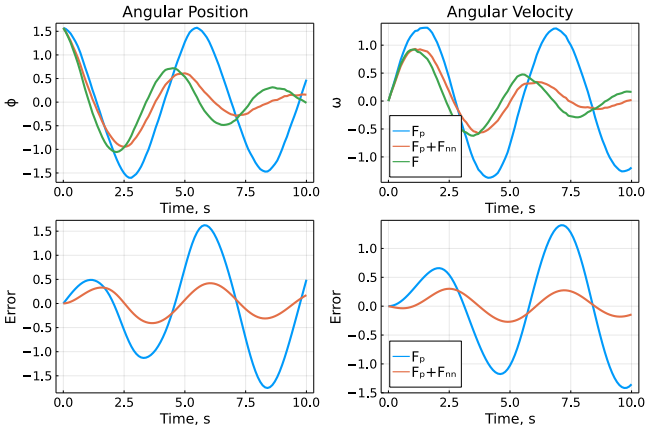

Fig. 1: (*Top row*) Comparison between the real trajectory (F) of the pendulum, the strictly physics-based model (F_p), and our physics-augmented neural network ($F_p + F_{nn}$). (*Bottom row*) Prediction errors calculated w.r.t. the real trajectory (F).

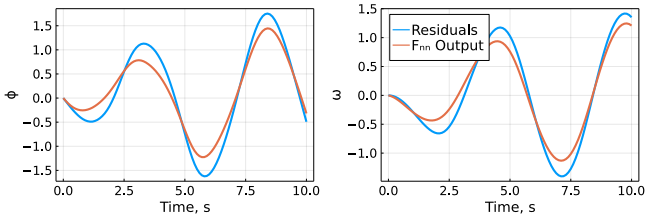

Fig. 2: Comparison of the residual dynamics (δ_k) with the neural network (F_{nn}) output that learns the residuals.

where ϕ is the angular position, ω the angular velocity, $\psi = 2$ the fundamental frequency, $\alpha = 0.4$ the attrition coefficient, $\Delta_t = 0.1$ the sampling time, and d and ν zero mean Gaussian random variables with covariance matrices $0.05^2 I$ and $0.01^2 I$.

The parametric representation or the physics-based model of the pendulum is assumed to not account for the damping and is given by

$$F_p(\phi, \omega, \psi) = \begin{bmatrix}\phi \\ \omega\end{bmatrix} + \int_t^{t+\Delta_t} \begin{bmatrix}\omega(\tau) \\ -\psi\phi(\tau)\end{bmatrix} d\tau. \quad \text{(15)}$$

To learn the unmodeled part of the dynamics, we use a neural network F_{nn} with a single hidden layer of width 50 and with ReLU as the first and identity as the second activation functions.

The optimal parameters are determined using (12a) and (12b) with a thousand trajectories of (14) with a time span of 1 second (and therefore 10 thousand data points) and each trajectory's initial condition drawn from a zero mean Gaussian variable with covariance matrix $0.6^2 I$. Once the parameters are learned, we compare the trajectories of the real model (14a) and the learned model given the same initial conditions and disturbances over a time span of 10 seconds. The results are shown in Figure 1.

The reconstructed trajectory from the learned dynamic system is able to represent the general behavior of the real system, as seen in the top row of Figure 1, while the trajec-

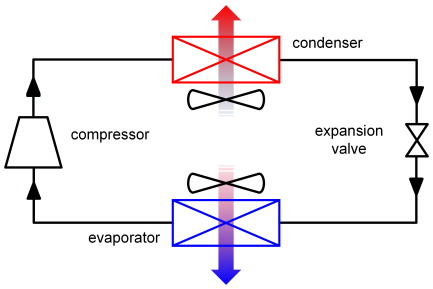

Fig. 3: Structure of a vapor compression cycle.

tory of physics-based model (F_p) deviates significantly from the real trajectory. The errors in angular position and velocity, as shown in bottom row of Figure 1, are significantly larger for the physics-based model (F_p) which does not account for the damping. The reconstructed trajectory has damping behavior, which would not be possible without the neural network. As the purpose of the neural network is to learn the residual dynamics not modeled in F_p, *i.e.,* the damping behavior of the system, we compare the neural network output to the residual in Figure 2. While the neural network output is close to the residual, the small difference between them can be attributed to noise and practical limitations of neural approximations from finite data. The neural network can be seen to learn from the estimated state trajectories despite the presence of these disturbances.

III. Application to Vapor Compression Cycles

Figure 3 illustrates a vapor compression cycle consisting of two refrigerant-to-air heat exchangers (HEXs) with variable speed fans, a variable-speed compressor, and a variable-position expansion device. This system transports thermal energy from the air passing through the evaporating HEX to the air passing through the condensing HEX via the refrigerant by using the latent heat of condensation and evaporation, where the compressor can operate efficiently at pressures for which the refrigerant evaporating temperature can be set lower than common occupied space temperatures, while the refrigerant condensing temperature can be set higher than common ambient temperatures.

A physics-based model of the system shown in Figure 3 was developed to describe its behavior. Since HEXs dominate the overall cycle dynamics, the HEX models were constructed from an index-1 system of differential algebraic equations (DAEs), while algebraic models were constructed for the compressor and expansion valve. The development of HEX models was facilitated by a series of simplifying assumptions, such as one-dimensional pipe flow, thermodynamic equilibrium in each discrete volume of the refrigerant pipe at each instant in time, negligible gravitational forces, and equality of the liquid and vapor phase velocities in the two-phase region.

A spline-based representation of the thermodynamic refrigerant properties [25] was utilized in the refrigerant pipe model that enforces the conservation of mass, momentum, and energy. The conservation laws expressed as nonlinear partial differential equations can be adapted to a finite control volume discretization into an arbitrary number volumes N_{vol} that employs a staggered-grid approach to maintain numerical stability. Increasing N_{vol} generally improves the accuracy of the model, but also increases its computational complexity. The airflow across each volume was modeled using an algebraic model, while an ordinary differential equation (ODE) model was constructed for the refrigerant pipe wall with one heat storage element. Each heat exchanger element thus comprises a refrigerant-side volume, a pipe wall element, and an air-side volume results in a set of index-1 DAE which is transformed in to a set of ordinary differential equations via the the Pantelides algorithm. The state vector x after index reduction contains the pressures (P_i), specific enthalpies (h_i), and temperatures (θ_i) for each discretized volume for the two HEXs, for a total of $3 \times 2 \times N_{vol}$ dimensions. The control inputs to the model are compressor speed and expansion valve position. For the purpose of this study, both of these control inputs are assumed to be piecewise constants. Further information on the HEX, compressor and expansion valve models is available in [5].

This model was implemented in the Julia programming language using ModelingToolkit.jl [26], which provides a framework that allows the models to be defined in a declarative context that can be used with other packages such as differential equation solvers and automatic differentiation tools. Further details on the model implementation can be found in [3].

A. Problem Setup

Epistemic and aleatoric uncertainty imposes practical limitations on the fidelity of models created using the results of Section II, which motivates this work in reducing the model mismatch by combining the physics-based model with a neural network. Our experimental setup uses a dual model approach. As a proxy for the real (and unknown) cycle model $F(\cdot)$, we create a model that discretizes the heat exchangers into 9 volumes each, for a total of 54 states, to produce a continuous time model $f_9(\cdot)$. We then discretize in time $f_9(\cdot)$ to obtain a "high-fidelity" discrete time model

$$x^+ = x + \int_t^{t+\Delta_t} f_9(x(\tau), u) d\tau + d = F(x, u) + d \quad (16)$$

where d is a zero mean random Gaussian disturbances. A separate physics-based model F_p, which is known to exhibit differences in behavior from the high-fidelity model, is then constructed by discretizing the heat exchangers into 3 volumes each, for a total of 18 states, to produce a continuous time model $f_3(\cdot)$. We then discretize in time f_3 to obtain a "low-fidelity" discrete time model

$$x^+ = x + \int_t^{t+\Delta_t} f_3(x(\tau), u) d\tau = F_p(x, u). \quad (17)$$

The integrals in (16) and (17) are calculated numerically because of the high-dimensional complex nonlinear nature of the model. This system is numerically quite stiff, thus, numerical solvers that can handle stiff problems are required, and the solver `QNDF` [27] is employed to propagate this model forward in time.

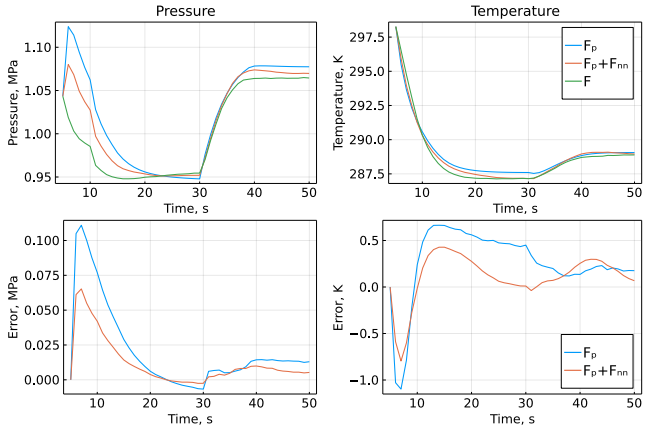

Fig. 4: Top row of plots shows comparison between the trajectory of the high fidelity VCC model (F), the low fidelity model (F_p) and the reconstructed using low fidelity physics-augmented neural networks (F_p+F_{nn}) for states of one of the finite volumes. The bottom row shows corresponding errors calculated with respect to high fidelity VCC model (F).

B. Numerical Results

In real world applications, cost and reliability considerations limit the availability of sensors in common vapor compression equipment. To incorporate this limitation in our problem, we assume that only 18 of the 54 states of the high fidelity system can be measured, with sensors that measure pressures, enthalpies and temperatures at select locations. The sensors are assumed to have some imprecisions, which is represented by an additive zero mean Gaussian noise.

The deep neural network component F_{nn} has five hidden layers, with ReLU activation functions. The input layer has 18 neurons associated with the states, 2 neurons associated with the controls, and 1 neuron for the time for a total of 21 input neurons. While neither $F(\cdot)$ nor $F_p(\cdot)$ are explicitly time-dependent on time, the control inputs are time-dependent and we obtained better results via its inclusion. The hidden layers have each 100 neurons. The output layer has 18 neurons. This deep neural network is implemented in Julia using Lux [28] and with automatic differentiation provided by Zygote [29].

We first generate 1000 initial conditions to build the dataset. We then run the high-fidelity model for a time span of 50 seconds, with sampling time (Δ_t) of 1 second, generating a total of 5×10^4 data points. The trajectories are divided into 600 for training, 300 for validation and 100 for testing. By using these limited measurements, this problem falls into the category of partial noisy state observation and process disturbances as described in Section II-C, for which we have to solve (12a) and (12b). Note that in this case there are no parameters θ for $F_p(\cdot)$ to be estimated. A more general architecture which includes it will be developed in future work.

Instead of explicitly solving (12a), we pose this as an estimation problem described in Section II-B, which we approximately solve using a constrained extended Kalman smoother (C-EKS) from [3] which is customized for VCC applications. C-EKS explicitly enforces state-constraints such as monotonic decrements in pressure in the direction of refrigerant flow in the system, which otherwise would not have been guaranteed to be satisfied during the estimation process. This also ensures that the residuals from which the neural network learns are consistent with the fundamental physics of the cycle. C-EKS can be viewed as a modified form of Rauch–Tung–Striebel (RTS) smoother that explicitly incorporates state-constraints during measurement updates using PDF truncation methods. Detailed equations and discussion on C-EKS are available in [3].

For training the neural network, which means solving (12b), we use the Adam optimizer on 6000 epochs. Every 50 epochs we test the parameters w on the validation set, and if there is an improvement in cost compared to the previous evaluation, we choose that value of w as the optimal. The total time for training is about 10 minutes on an Nvidia Titan X desktop GPU.

In order to compare the quality of our reconstruction, for each trajectory $x_k^{(n)}$ in the training set with initial condition $x_0^{(n)}$ and disturbances $d_k^{(n)}$, we generate a reconstructed trajectory

$$\tilde{x}_{k+1}^{(n)} = F_p(\tilde{x}_k^{(n)}, u_k) + F_{nn}(\tilde{x}_k^{(n)}, u_k, w^*) + d_k^{(n)}$$

using the same initial condition $x_0^{(n)}$ and disturbances $d_k^{(n)}$.

The top row of Figure 4 shows the trajectories of pressure, specific enthalpy, and temperature in the volume at the inlet to the evaporating HEX for one of the trajectories for the high fidelity model (F), the low fidelity model (F_p) and the reconstructed states ($F_p + F_{nn}$). The corresponding errors calculated with respect to the high fidelity model (F) are shown in the bottom row of Figure 4. The reconstructed state is better at generating a trajectory that approximates the high fidelity model than just using the low fidelity model. In particular, the reduced maximum amplitude of the error for all three variables suggests that the reconstructed states represent a better description of the underlying system behavior than the physics-based model F_p alone.

A comparison of the simulation time to run both the high-fidelity model F and the low-fidelity model F_p also provides additional insight into the application of these modeling methods. Whereas it takes 16 sec to integrate F from 0 to 50 sec, the low-fidelity model can be solved over the same time interval in 10 sec. This suggests that the tradeoff between forecast accuracy and computational complexity in models can be addressed in part by using these augmented modeling techniques. Rather than pay a high computational price by using a high-fidelity model for prediction, it may be possible to instead use a low-fidelity physics-augmented neural network to obtain state-predictions of acceptable quality for a lower computational cost.

The point-wise normalized error of an estimated trajectory $x_k^{(n)}$ calculated with respect to the high-fidelity trajectory $\bar{x}_k^{(n)}$ is given as $\varepsilon_k^{(n)} = (x_k^{(n)} - \bar{x}_k^{(n)})/(\bar{x}_k^{(n)})$ where the vector division is interpreted element-wise. The error statistics

TABLE I: Error statistics of the low fidelity (F_p) and the reconstructed ($F_p + F_{nn}$) trajectories.

Error	F_p	$F_p + F_{nn}$
Mean	3.8%	2.3%
Standard deviation	6.8%	3.5%
Maximum	53.9%	35.6%

calculated over each state, time step, and trajectory for the low fidelity (F_p) and the reconstructed (F_p+F_{nn}) trajectories are shown in Table I. The learned model with neural network demonstrates overall better accuracy than the low-fidelity model. It should be noted that the performance of hybrid model F_p+F_{nn} is limited by the low-fidelity model F_p which is used for state estimation and residual calculations in (12a). However, as seen from Table I, when the neural network is included, the prediction accuracy is improved about by 40-50% (in terms of mean and standard deviation) with respect to low-fidelity model. The overall accuracy of the hybrid model $F_p + F_{nn}$ can be further improved if the parameters of F_p are also optimized, which has not been an aspect of this study and is subject to future work.

IV. Conclusions

In this article, we developed and demonstrated the effectiveness of a framework to augment physics-based models of complex physical systems such as vapor compression cycles (VCC) with neural networks to learn unmodeled dynamics and achieve better prediction capabilities using partial noisy state-observations. The problem of jointly estimating the VCC model parameters and training neural network was decoupled in to two optimizations that were solved, respectively, using state-constrained Kalman smoothing algorithms and an Adam optimizer. Numerical results showed that an approximate low-fidelity VCC model, when augmented with a neural network, was able to achieve improved prediction accuracy which can be leveraged to improve overall performance of the system.

References

[1] IEA, "The future of cooling," tech. rep., International Energy Agency, Paris, 2018.

[2] S. Bortoff, P. Schwerdtner, C. Danielson, and S. Di Cairano, "H-infinity loop-shaped model predictive control with heat pump application," in *18th European Control Conference*, pp. 2386–2393, 2019.

[3] V. M. Deshpande, C. R. Laughman, Y. Ma, and C. Rackauckas, "Constrained smoothers for state estimation of vapor compression cycles," in *2022 American Control Conference*, pp. 2333–2340, 2022.

[4] C. Vering, S. Borges, D. Coakley, H. Kruetzfeldt, P. Mehrfeld, and D. Müller, "Digital twin design with on-line calibration for HVAC systems in buildings," in *Proceedings of Building Simulation 2021: 17th Conference of IBPSA*, vol. 17 of *Building Simulation*, (Bruges, Belgium), pp. 2938–2945, IBPSA, September 2021.

[5] H. Qiao, V. Aute, and R. Radermacher, "Transient modeling of a flash tank vapor injection heat pump system–Part I: Model development," *International Journal of Refrigeration*, vol. 49, pp. 169–182, 2015.

[6] C. R. Laughman and H. Qiao, "On closure relations for dynamic vapor compression cycle models," in *American Modelica Conference*, Oct. 2018.

[7] S. L. Brunton, J. L. Proctor, and J. N. Kutz, "Discovering governing equations from data by sparse identification of nonlinear dynamical systems," *Proceedings of the National Academy of Sciences*, vol. 113, no. 15, pp. 3932–3937, 2016.

[8] V. M. Deshpande and R. Bhattacharya, "Surrogate modeling of dynamics from sparse data using maximum entropy basis functions," in *2020 American Control Conference (ACC)*, pp. 4046–4051, 2020.

[9] K. S. Narendra and K. Parthasarathy, "Neural networks and dynamical systems," *International Journal of Approximate Reasoning*, vol. 6, no. 2, pp. 109–131, 1992.

[10] Y. Yin, V. L. Guen, J. Dona, E. de Bezenac, I. Ayed, N. Thome, and P. Gallinari, "Augmenting Physical Models with Deep Networks for Complex Dynamics Forecasting," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2021, p. 124012, Dec. 2021. arXiv:2010.04456 [cs, stat].

[11] J. Dona, M. Déchelle, P. Gallinari, and M. Levy, "Constrained Physical-Statistics Models for Dynamical System Identification and Prediction," in *Tenth International Conference on Learning Representations*, Sept. 2021.

[12] M. Buisson-Fenet, V. Morgenthaler, S. Trimpe, and F. Di Meglio, "Learning dynamics from partial observations with structured neural ODEs," May 2022. arXiv:2205.12550.

[13] G. Shi, X. Shi, M. O'Connell, R. Yu, K. Azizzadenesheli, A. Anandkumar, Y. Yue, and S.-J. Chung, "Neural lander: Stable drone landing control using learned dynamics," in *2019 International Conference on Robotics and Automation (ICRA)*, pp. 9784–9790, IEEE, 2019.

[14] M. Saveriano, Y. Yin, P. Falco, and D. Lee, "Data-efficient control policy search using residual dynamics learning," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4709–4715, IEEE, 2017.

[15] K. Kaheman, E. Kaiser, B. Strom, J. N. Kutz, and S. L. Brunton, "Learning discrepancy models from experimental data," *arXiv preprint arXiv:1909.08574*, 2019.

[16] A. Farchi, P. Laloyaux, M. Bonavita, and M. Bocquet, "Using machine learning to correct model error in data assimilation and forecast applications," *Quarterly Journal of the Royal Meteorological Society*, vol. 147, no. 739, pp. 3067–3084, 2021.

[17] Y. D. Zhong, B. Dey, and A. Chakraborty, "Dissipative SymODEN: Encoding Hamiltonian Dynamics with Dissipation and Control into Deep Learning," 2020.

[18] J. Drgoňa, A. R. Tuor, V. Chandan, and D. L. Vrabie, "Physics-constrained deep learning of multi-zone building thermal dynamics," *Energy and Buildings*, vol. 243, p. 110992, 2021.

[19] Y. Peng, A. Rysanek, Z. Nagy, and A. Schlüter, "Using machine learning techniques for occupancy-prediction-based cooling control in office buildings," *Applied Energy*, vol. 211, pp. 1343–1358, 2018.

[20] C. Bhattacharya, A. Chakrabarty, C. R. Laughman, and H. Qiao, "CNN-GRU: Efficient Learning of Thermo-Fluid Dynamical Systems with Convolutions and Recurrence," in *To appear, Proc. Modeling Estimation and Control Conference,, New Jersey, NJ, USA, 2022(T)*, 2022.

[21] D. Jani, M. Mishra, and P. Sahoo, "Performance prediction of solid desiccant – vapor compression hybrid air-conditioning system using artificial neural network," *Energy*, vol. 103, pp. 618–629, 2016.

[22] J. Gill and J. Singh, "Use of artificial neural network approach for depicting mass flow rate of R134a/LPG refrigerant through straight and helical coiled adiabatic capillary tubes of vapor compression refrigeration system," *International Journal of Refrigeration*, vol. 86, pp. 228–238, 2018.

[23] M. Raissi, P. Perdikaris, and G. E. Karniadakis, "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations," *Journal of Computational physics*, vol. 378, pp. 686–707, 2019.

[24] M. Asch, M. Bocquet, and M. Nodet, *Data assimilation: methods, algorithms, and applications*. No. 11 in Fundamentals of algorithms, Philadelphia: SIAM, Society for Industrial and Applied Mathematics, 2016. OCLC: 992490263.

[25] C. Laughman and H. Qiao, "Patch-based thermodynamic property models for the subcritical region," in *International Refrigeration and Air-Conditioning Conference at Purdue*, pp. 1–10, 2021. Paper 2258.

[26] Y. Ma, S. Gowda, R. Anantharaman, C. R. Laughman, V. Shah, and C. Rackauckas, "ModelingToolkit: A composable graph transformation system for equation-based modeling," 2021.

[27] L. F. Shampine and M. W. Reichelt, "The MATLAB ODE suite," *SIAM Journal on Scientific Computing*, vol. 18, no. 1, pp. 1–22, 1997.

[28] A. Pal, "Lux: Explicit parameterization of deep neural networks in julia." `https://github.com/avik-pal/Lux.jl/`, 2022.

[29] M. Innes, "Don't Unroll Adjoint: Differentiating SSA-Form Programs," *CoRR*, vol. abs/1810.07951, 2018.