# Physics-informed Recurrent Neural Networks for the identification of a generic energy buffer system

Manu Lahariya[1], Farzaneh Karami[2], Chris Develder[1], Guillaume Crevecoeur[2]

1. IDLab, Ghent University, Technologiepark-Zwijnaarde 126, Ghent, Belgium – imec
E-mail: manu.lahariya@ugent.be,chris.develder@ugent.be

2. Dept. Electromechanical, Systems and Metal Engineering, Ghent University Core lab EEDT-DC, Flanders Make
Technologiepark-Zwijnaarde 131, B-9052 Gent, Belgium
E-mail: karami.farzaneh@ugent.be, guillaume.crevecoeur@ugent.be

**Abstract:** Energy storage is ubiquitous in industrial processes and comes in many forms such as material, chemical, electromechanical buffers. System identification of such energy buffers demands proper estimation/prediction of their physical quantities and unknown parameters. Once these parameters are determined, the identified model can be employed to predict the industrial process dynamics, which finally assist to build efficient control for these processes. This paper proposes physics-informed neural networks-based grey-box modeling methods for the identification of energy buffers. The underlying system dynamics are enforced on the neural network structure to ensure that the identified grey-box model follows the approximate physics. We define two novel grey-box models based on simple and recurrent neural network architectures and test these models for a generic energy buffer. Performance and training time for the proposed grey-box models are compared against a black-box baseline model. Results confirm that imposing the dynamic system's physics on the network improves the performance, and utilizing a recurrent architecture leads to a further improvement.

**Key Words:** Recurrent Neural Networks, Physics-informed Neural Networks, PyLSTM, System identification

## 1 Introduction

Energy market fluctuations can cause problems in the reliable operation of industrial processes such as petro(chemical) and steel production. Disturbances need to be mitigated as systems operate in a reliable and stable manner. Effective control strategies that offer to industrial processes the needed flexibility can help to maximize energy efficiency and ultimately effective grid operation. Operational flexibility in an industrial process is defined as the ability of these processes to adapt to changes in the energy market quickly and with little financial effort. Control engineers rely on parameterized models as these provide the insights on the effect of settings on the dynamic behavior and operation of the systems. When considering flexible energy systems, identification of the system parameters is thus essential step in the control design. The dynamics can be formalized by a generic energy buffer model [1]. The generic buffer is a virtual battery that outlines the relationship between the control input and system parameter. This generic buffer formulation can be used to describe multiple industrial processes such as an evaporative cooling tower (basin temperature as flexible parameter), wind turbines, and Photovoltaics (PV) units.

System identification allows to build to build mathematical models to apprehend a system's behavior by relating the inputs to its outputs. It can assist developing effective control, making pricing/cost decisions etc. Classical dynamic system modeling approaches are based on knowledge of the physical phenomena to define this relationship, resulting in partial/ordinary differential equations (PDEs/ODEs). Such dynamic system models are often referred to as white-box models. White box models provide the ability that control engineers can make sure that the industrial process is controlled in a flexible manner. However, the complexity and large scale nature of industrial processes hinders using only white-box models, i.e., the limited set of parameters in the models fail to fully align with real-world processes [2]. Data-driven approaches have attracted researchers' attention when it comes to analyzing or understanding complex processes. Machine learning based system identification allows to learn the relationship between control inputs and unknown parameters [3]. Such black-box models however lack the physical understanding of the system and can result in outputs that are physically improbable as they fail to extrapolate well. A good example is the use of an adaptive neuro-fuzzy inference for predicting basin temperature in an evaporate cooling tower [4], where it is noted that white-box models tend to perform better. However, these black-box models have been studied for specific industrial applications and not for the generic buffer formulation.

Grey-box modeling is a more attractive approach than black-box modeling because it incorporates the system's underlying dynamics in the data-driven machine learning model. They usually outperform the black-box models in predicting the future state of the system. Traditional grey-box models for system identification are based on data-driven techniques to integrate the physics of the process [5]. In recent years, multiple physics-informed modeling methods have been studied, including Lagrangian neural networks [6], Bayesian techniques [7] and physics-guided recurrent neural networks [8, 9]. Physics-informed neural networks [10] have been tested for classical problems (e.g., fluid dynamics, reaction-diffusion systems and such) but do not explore the highly complicated nature of an industrial process. Training these neural network-based models requires a large amount of data. In an industrial context, mostly only scarce data is available due to not logging the data due to lack of proper infrastructure. Furthermore, the collected sensor-based data has not yet been fully exploited to understand the industrial process.

We define two novel grey-box models for system identification in a generic industrial process, namely physics-informed neural networks (PyNN) and physics-informed long-short term memory networks (PyLSTM). These models enforce the system's dynamics onto the network architecture compared to a black-box neural network model (NN) that follows a free structure. We employ a recurrent architecture in PyLSTM that is highly beneficial for modelling the time-varying processes. In this paper we aim to evaluate the potential of these grey-box models by implementing and testing for a simplified generic energy buffer. Furthermore, we compare these models against the baseline neural network (NN) model. To assess our approach performance, we compare our models' output against simulated data for a generic buffer. The present work has, therefore, the following two contributions. i) Define the architecture for physics-informed neural networks for generic buffer system identification, ii) Performance comparison between the proposed grey-box models and the traditional data-driven model.

## 2 Generic Buffer System

Generic buffers can be considered as virtual batteries that provide operational flexibility in the system [1]. Mapping industrial processes to a simple dynamic linear model, a generic buffer, also enables identifying their hidden (non-identified) operational flexibility by considering input and output resources to/from each process. A generic buffer model constructed by representing the process in the form of a power node [11] can be used to identify the parameter that provides flexibility in the system. Furthermore, this generic buffer brings the potential of having a more customized objective function (e.g., maximize flexibility, minimize power consumption and such) during industrial process control design.

An industrial process that provides flexibility can be represented as a virtual battery as shown in Figure 1. The power fed into the process and that comes out of the process are given by $P_{in}$ and $P_{out}$. The state of charge is represented by *E*. For example, in the case of a storage tank, $E$ can be represented using the current state of storage ($m$ represents how much the tank is filled, Equation 2), where *K* is efficiency rate. The system's dynamic equation is given by Equation 1 and we can write the constraint on $E$ as in Equation 3. Representing a process in this format provides an opportunity for identifying the system parameter $E$ and developing control algorithms based on this identified parameter.

$$\frac{\partial m}{\partial t} = K(P_{out} - P_{in}) \tag{1}$$

$$E = \frac{m - m_{min}}{K} \tag{2}$$

$$E_{min} \leq E \leq E_{max} \tag{3}$$

The physics-informed models identify the system parameter ($m$) that offers operational flexibility in this generic buffer. This identified parameter can be employed to calculate the current state of charge for this virtual battery of the process. As a case study for building and testing our models, we consider the case of an RC-circuit. An RC-circuit can be considered as a virtual battery, where the voltage across the capacitor is the system parameter that provides flexibility.

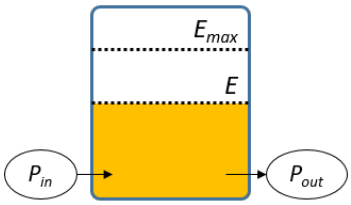


Figure 1: figure
Generic buffer
(Used to represent industrial processes in form of virtual battery)

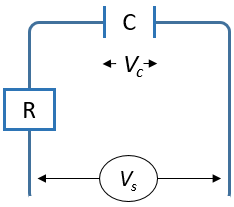


Figure 2: figure
RC-circuit as a virtual battery
($V_s$: Input voltage, $V_c$: Capacitor voltage)

### 2.1 RC-Circuit

We consider a simplified example of an RC-circuit to investigate the feasibility of applying physics-informed neural networks as (i) it is representative of the generic buffer model, (ii) we can easily simulate/obtain data for experimental purposes. A RC-circuit (Figure 2) can be represented as a generic buffer using Equation 4, where $V_s$ denotes the input voltage, $V_c$ is the voltage across the capacitor.

$$\frac{\partial V_c}{\partial t} = \frac{V_s - V_c}{RC} \tag{4}$$

The underlying dynamic of the RC-circuit is very similar to a generic buffer, as evident from the similar representations of Equation 7 and Equation 1 (dynamic equation of a virtual battery). The system parameter ($m$) is the voltage across the capacitor ($V_c$). The control parameters are the resistance ($R$), capacitance ($C$) and the voltage supplied ($V_s$), which means our models need to approximate a single output parameter $V_c$ using multiple input parameters. Such a structure helps in evaluating the efficacy of our models in the case of a standard generic buffer. Furthermore, we can simulate reliable data for RC-circuit step response. Real-world industrial data is challenging to acquire and often deficient for such analysis, and choosing to test our models on RC-circuit solves that problem.

## 3 Physics-Informed Networks

For grey-box models, we use two model architectures: physics-informed neural networks (PyNN) based on previous work [10] and physics-informed long short term memory (PyLSTM) networks. Both models' underlying idea is to enforce the governing physics in the neural network architecture using the system's dynamic equations. A dynamic system can be represented using Equation 5, where $m$ is the system response and $H$ is a non-linear operator. We assume that the PDE representing the system can be rewritten as in Equation 6. For and RC-circuit, we can rewrite Equation 4 as Equation 7 where $m = V_c$ is the system response that needs to be characterized.

$$\frac{\partial m}{\partial t} + H(m;\lambda) = 0 \tag{5}$$

$$f = \frac{\partial m}{\partial t} + H(m;\lambda) \tag{6}$$

$$f = \frac{\partial V_c}{\partial t} - \frac{V_s - V_c}{RC} \tag{7}$$

*Black-box model*: A neural network can be used to approximate the time-varying relationship between the control inputs and system response. System response approximated by neural network is represented by $\widehat{m}^{NN}$ and depends on the inputs $I$ (Equation 8, where $I_i$ are the inputs for $i^{th}$ sample). In case of RC-circuit, $\widehat{V}_c^{NN}$ depends on the control inputs $(R, C, V_s)$ and time $t$ when the sample was recorded (Equation 9). The weights of the trained neural network are represented by $\theta$.

$$\widehat{m}_i^{NN} = NN(I_i;\theta) \tag{8}$$

$$\widehat{V}_{c,i}^{NN} = NN(R_i, C_i, V_{s,i}, t_i;\theta) \tag{9}$$

In the following sections, architectures of two grey-box neural networks are explained that enforce the dynamics of an RC-circuit in a neural network architecture.

### 3.1 Physics-Informed Neural Networks (PyNN)

Figure 3 depicts the architecture of PyNN, where $I_i$ corresponds to the input for $i^{th}$ sample. Estimated value of parameter *m*, represented by $\widehat{m}^{PyNN}$, is characterized by Equation 10. After calculating $\widehat{m}^{PyNN}$, its time derivative is calculated using auto differentiation and non-linear operator $H$ is applied to it to calculate $\widehat{f}^{PyNN}$. Time is fed as an input ($I_i$) to the neural network. This introduces the temporal component in the model and helps in learning how *m* evolves with time. In case of RC- circuit, the values $\widehat{V}_{c,i}^{PyNN}$ and $\widehat{f}_i^{PyNN}$ are calculated according to Equation 11 and Equation 12.

$$\widehat{m}_i^{PyNN} = PyNN(I_i;\theta) \tag{10}$$

$$\widehat{V}_{c,i}^{PyNN} = PyNN(R_i, C_i, V_{s,i}, t_i;\theta) \tag{11}$$

$$\widehat{f}_i^{PyNN} = \left(\frac{\partial \widehat{V}_c^{PyNN}}{\partial t}\right)_i - \frac{V_{s,i} - \widehat{V}_{c,i}^{PyNN}}{R_i C_i} \tag{12}$$

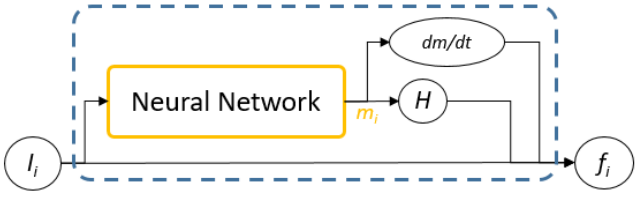


Figure 3: figure

[Physics-informed NN architecture]Physics-informed NN architecture. Inputs ($I$) for $i^{th}$ sample are used to model both $m$ and $f$.

### 3.2 Physics-Informed LSTM Networks (PyLSTM)

This section outlines a modified version of PyNN, where we use a long short term memory (LSTM) to model the data. LSTM networks are a type of recurrent neural networks (RNN), which are used for temporal or sequential modeling where the output of a step depends on the output of past steps. Each cell of RNNs returns two parameters, output and state, where the state represents the cell's state and is used as an input for the next step. LSTM cells have forgotten and update gates, based on which they update the state of the cell [12]. To make a prediction at step *i* we feed the data of past *i-k* steps, such that the $\widehat{m}^{PyLSTM}$ is characterized by Equation 13. We can choose *k* based on the system. We estimate the time derivative of *m* using finite difference based filtering. Figure 4 describes the architecture of PyLSTM. In case of RC-circuit, $\widehat{V}_{c,i}^{PyLSTM}$ is described by Equation 14, and $\widehat{f}_i$ is estimated similar to PyNN (Equation 12).

$$\widehat{m}_i^{PyLSTM} = PyLSTM(I_i, I_{i-1}, \ldots, I_{i-k};\theta) \tag{13}$$

$$\widehat{V}_{c,i}^{PyLSTM} = PyLSTM(R_i, C_i, V_{s,i}, t_i, \ldots \ldots, R_{i-k}, C_{i-k}, V_{s,i-k}, t_{i-k};\theta) \tag{14}$$

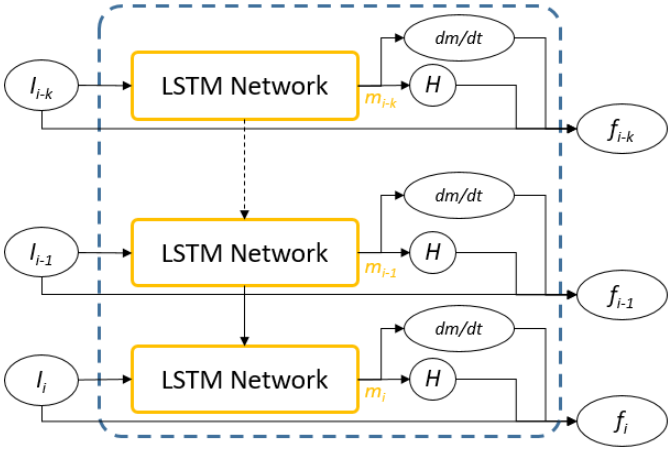


Figure 4: figure

Physics-informed LSTM architecture.

### 3.3 Training

Grey-box models (PyNN and PyLSTM) are trained by minimizing the error incurred in estimating system response ($e_m = m - \widehat{m}$) and error incurred in enforcing the dynamics of system ($e_m = f - \widehat{f}$). Integrating the error of $f$ ensures that the dynamic system's physics is enforced on the network. Physics-informed LSTM are trained using the 'teacher-forcing' algorithm introduced in [13]. In this training algorithm, the true value of output at $(i-1)^{th}$ step is used as input to the network at $i^{th}$ step, rather than using the network's output from the previous step.

A loss function defined as the total mean squared error (MSE), which is the sum of MSE of estimating $m$ and MSE of estimating $f$, is used previously to train physics-informed neural networks [10]. We extend this loss function by including L2 regularization to ensure a robust training of

the recurrent neural networks, which are prone to overfitting when trained 'teacher-forcing' algorithm. In Equation 15, we give the loss in the network with weights *W*. Total number of observations in a batch is represented by *N*, $e_{m,i}$ and $e_{f,i}$ are errors in estimating $m$ and $f$ respectively, and the last term is L2 regularization. Parameter $\lambda$ controls the importance of the regularization term.

$$\begin{aligned} L(W) &= MSE_m + MSE_f + \Omega(W) \\ &= \frac{1}{N}\sum_{i=1}^{N}(e_{m,i}^2 + e_{f,i}^2 + \lambda|W_i|^2) \end{aligned} \tag{15}$$

## 4 Experimental design

In the following paragraphs, we outline the data simulation for RC-circuit, training of the baseline neural networks and physics-informed networks, and the evaluation metrics.

**Data Simulation:** The step response of a RC-circuit can be represented using the Equation 7, where resistance (R), capacitance (C) and input voltage ($V_s$) are the system control parameters. The voltage across the capacitor ($V_c$) is the system parameter that represents flexibility and needs to be identified. Data is simulated for 5000 seconds (100 observations/second) based on randomly chosen values of $V_s$, R and C ($V_s \in \{0, 1, 2, 3\}V$; R $\in \{1, 2, 3\}\Omega$; C $\in \{1, 2, 3\}F$). Each sample also records the time (s) referred as timestep. Gaussian noise is added to the simulated $V_c$ to get a noisy observation $V_c^n$ that represents the real world scenario. This noisy observation is used to train the models.

**Model configuration:** Baseline neural networks (NN) and physic-based neural networks (PyNN) take the inputs at timestep *i* to make predictions at timestep *i*. Both architectures are fully connected networks with two hidden layers with 16 neurons in each layer. We build a Physics-informed LSTM (PyLSTM) with the length *k* = 50 steps to make predictions for the next step. Two hidden layers of 16 LSTM cells each are utilized. While training, 'teacher-forcing' is used to train the LSTM network at each step. While making predictions with PyLSTM, we feed the data for the last $k$ steps to make the prediction. Training and evaluation are performed using Intel Xeon E5645 and a single four-core E3-1220v3 (3.1GHz) with 16GB RAM.

**Evaluation:** We do a walk forward validation with a moving window, where a validation set is selected for a fixed window, and this window is moved ahead in time recursively to create the new validation sets. This validation is conducted because the data is time-series, and we can compare the performance based on the same training and testing set size for all models. The training set consists of data for 50K timesteps (500 s) and we evaluate our model on the next 5K timesteps (50 s). We do this for a total of 5 validation sets.

We calculate the absolute errors for all models based on short-term predictions (2 s, 200 time steps) and long-term predictions (50 s, 5000 time steps) to measure the efficacy. Absolute error is defined in Equation 16, where $\widehat{V}_c^M$ represents the values predicted by model $M \in \{NN, PyNN, PyLSTM\}$.

$$|e_i|_M = |V_{c,i} - \widehat{V}_{c,i}^M| \tag{16}$$

## 5 Numerical results

Figure 5 shows the absolute errors for all models for short-term and long-term predictions. Each box is constructed from absolute errors of all predictions in the prediction horizon, for example, for long-term (50 s), each box is made from 5K absolute errors. Average values are calculated for all validation sets. We can see that PyLSTM outperforms both models. In predictions for the next 2 seconds, PyNN performs better than the baseline neural network NN model because it has information on the system's dynamics. To check if the absolute errors for models are statistically different, we perform a two-tailed t-test ( Figure 5). We can see that even in long-term predictions, PyLSTM is statistically better from the baseline neural network (NN).

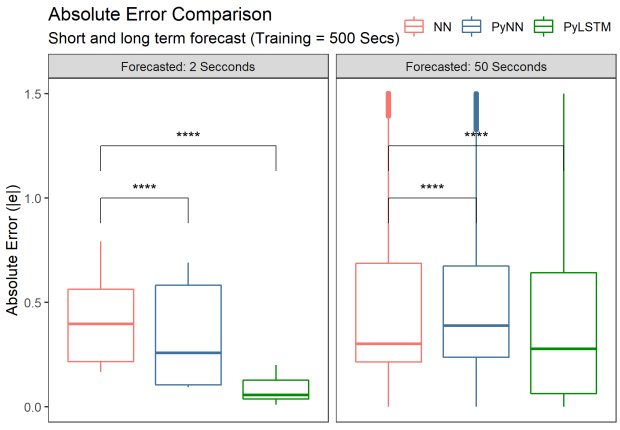


Figure 5: figure

Absolute error distribution for short-term predictions (2 s) and long-term predictions (50 s). (t-test p-Values are represented using **** : p-Value ¡ 0.0001)

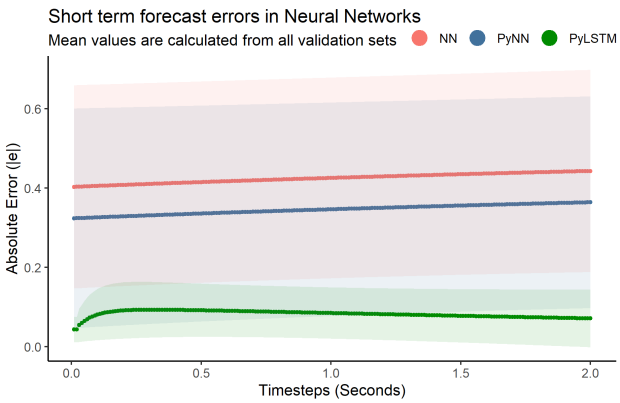


Figure 6: figure

Absolute error for short-term predictions. Average values are calculated for all validation sets (shaded area represents 25 and 75 percentile)

We also provide all models' absolute errors for each time step in Figure 6 for short-term predictions. PyLSTM seems'

to be performing better than both the other models. Adding information about the dynamics of the systems improves the performance of neural networks, but using a LSTM increases the performance even further with less than 5% error in short-term predictions. We also see that the variance in errors in the case of PyLSTM is smaller compared to the other two models. The worst performing validation set results in the case of PyLSTM which is still better than the best performing PyNN.

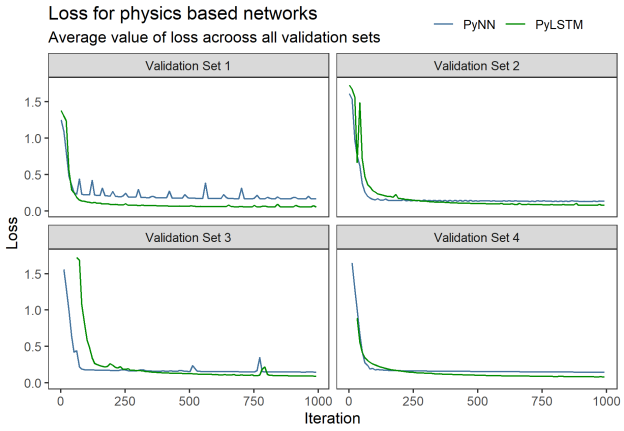


Figure 7: figure

Loss per iteration for physics-informed models (Loss is the sum of mean square error in $m$ and $f$, Equation 15)

Figure 7 shows the loss curves for physics-informed models for 4 validation sets (we notice similar curves for other validation sets). The loss for PyNN becomes stable after around 100 iterations in all cases, and it is optimized faster than PyLSTM. Physics-informed LSTM takes a large number of iterations to optimize, where the loss keeps decreasing even after 500 iterations. This also suggests that using more data will result in a better model in the case of PyLSTM. Training time for PyLSTM for each iteration is higher than PyNN, which happens because we run more operations while training a recurrent neural network compared to the standard neural networks.

## 6 Conclusions

Industrial processes depend on accurate system identification to design effective control polices, mitigating future disturbances and and reliable operation. It is also crucial in forecasting parameters that assist in making pricing/cost decisions. In this paper, we present two architectures of physics-informed neural networks (PyNN and PyLSTM), that can be employed for system identification in dynamic systems. One such dynamic system that is representative of a generic industrial process is used to evaluate and compare the performance of these models.

We notice significant improvement in system identification in grey-box models (PyNN and PyLSTM) which are not only optimized for minimum output prediction error, but also incorporate adherence to physical model constraints (i.e., ODE/PDE) compared to the black-box model (NN). Furthermore, the recurrent neural network architecture based PyLSTM outperform even the grey-box PyNN, and we receive the best performance for PyLSTM such that short-term prediction errors for this network are less than 5%. We also see that PyLSTM keeps on learning even after loss for PyNN becomes constant. For predictions in the immediate future of the training period (i.e., short-term predictions), PyLSTM should be used as they outperform other models.

This study was a initial proof-of-concept, demonstrating the potential of PyLSTM for system identification in dynamic systems. Future work will include validation for (i) real-world industrial processes, and (ii) multiple input and multiple output systems compared to single output system that we tested. Finally, as mentioned, we need to come up with a DL-based control by employing PyLSTM.

## Acknowledgements

Part of the research leading to these results has received funding from Agentschap Innoveren & Ondernemen (VLAIO) as part of the Strategic Basic Research (SBO) program under the InduFlexControl project and the European Union's Horizon 2020 research and innovation program for the project BIGG (grant agreement no. 957047)

## References

[1] A. Ulbig and G. Andersson, "Analyzing operational flexibility of electric power systems," *International Journal of Electrical Power Energy Systems*, vol. 72, pp. 155 – 164, 2015. The Special Issue for 18th Power Systems Computation Conference.

[2] T. Chai, S. J. Qin, and H. Wang, "Optimal operational control for complex industrial processes," *Annual Reviews in Control*, vol. 38, no. 1, pp. 81–92, 2014.

[3] A. Chiuso and G. Pillonetto, "System identification: A machine learning perspective," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 2, no. 1, pp. 281–304, 2019.

[4] J. Baetens, G. Van Eetvelde, G. Lemmens, N. Kayedpour, J. De Kooning, and L. Vandevelde, "Thermal performance evaluation of an induced draft evaporative cooling system through adaptive neuro-fuzzy interference system (anfis) model and mathematical model," *ENERGIES*, vol. 12, no. 13, p. 17, 2019.

[5] T. P. Bohlin, *Practical Grey-Box Process Identification: Theory and Applications*. Springer Publishing Company, Incorporated, 1st ed., 2010.

[6] M. Lutter, C. Ritter, and J. Peters, "Deep lagrangian networks: Using physics as model prior for deep learning," *ArXiv*, vol. abs/1907.04490, 2019.

[7] H. J. Tulleken, "Grey-box modelling and identification using physical knowledge and bayesian techniques," *Automatica*, vol. 29, no. 2, pp. 285–308, 1993.

[8] N. Mohajerin and S. L. Waslander, "Multistep prediction of dynamic systems with recurrent neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 11, pp. 3370–3383, 2019.

[9] X. Jia, J. Willard, A. Karpatne, J. Read, J. Zwart, M. Steinbach, and V. Kumar, "Physics guided rnns for modeling dynamical systems: A case study in simulating lake temperature profiles," in *SIAM International Conference on Data Mining, SDM 2019*, SIAM International Conference on Data Mining,

SDM 2019, pp. 558–566, Society for Industrial and Applied Mathematics Publications, 2019.

[10] 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.

[11] K. Heussen, S. Koch, A. Ulbig, and G. Andersson, "Energy storage in power system operation: The power nodes modeling framework," in *2010 IEEE PES Innovative Smart Grid Technologies Conference Europe (ISGT Europe)*, pp. 1–8, 2010.

[12] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, pp. 1735–80, 12 1997.

[13] R. J. Williams and D. Zipser, "A learning algorithm for continually running fully recurrent neural networks," *Neural Computation*, vol. 1, no. 2, pp. 270–280, 1989.