Composable and Reusable Neural Surrogates to Predict System Response of Causal Model Components

Ranjan Anantharaman [1], Anas Abdelrehim [2], Francesco Martinuzzi [2,3,4], Sharan Yalburgi [2], Elliot Saba [2], Keno Fischer [2], Glen Hertz [2], Pepijn de Vos [2], Chris Laughman [5], Yingbo Ma [2], Viral Shah [2], Alan Edelman [1,2], Chris Rackauckas [2]

[1] Massachusetts Institute of Technology, [2] Julia Computing Inc., [3] Remote Sensing Centre for Earth System Research, Leipzig University, [4] Center for Scalable Data Analytics and Artificial Intelligence, Leipzig University, [5] Mitsubishi Electric Research Lab

Abstract

Surrogate models, or machine learning based emulators of simulators, have been shown to be a powerful tool for accelerating simulations. However, capturing the system response of general nonlinear systems is still an open area of investigation. In this paper we propose a new surrogate architecture which is capable of capturing the input/output response of causal models to automatically replace large aspects of block model diagrams with neural-accelerated forms. We denote this technique the Nonlinear Response Continuous-Time Echo State Network (NR-CTESN) and describe a training mechanism for it to accurately predict the simulation response to exogenous inputs. We then describe a science-guided or physics-informed surrogate architecture based on Cellular Neural Networks to enable the NR-CTESN to accurately reproduce discontinuous output signals. We demonstrate this architecture on an inverter circuit and a Sky130 Digital to Analog Converter (DAC), showcasing a 9x and 300x acceleration of the respective simulations. These results showcase that the NR-CTESN can learn emulate the behavior of components within composable modeling frameworks and thus be reused in new applications without requiring retraining. Together this showcases a machine learning technique that can be used to generate nonlinear model order reductions of model components in SPICE simulators, Functional Markup Interface (FMI) representations of causal model components, and beyond.

Keywords: nonlinear system response, surrogate modeling, causal modeling, composable abstractions

Introduction

Optimization of system designs commonly requires thousands to millions of expensive model simulations in order to identify to global or local minimums (Du et al. 2018; Yildiz, Abderazek, and Mirjalili 2020). To address this computational cost, reduced order and surrogate models (Willard et al. 2020) have become standard practice in many domains to accelerate simulations (Chatterjee, Chakraborty, and Chowdhury 2019; Han et al. 2017). Surrogates, and in particular physics-informed surrogates (White et al. 2019), are popular as stand-ins for full order simulations. They are also widely used in optimization (Stern, Song, and Work 2017), uncertainty quantification (Tripathy and Bilionis 2018), and control design (Peitz and Dellnitz 2018).

Training surrogate models presents two difficulties. First, many systems used in engineering practice exhibit stiffness and have dynamics that are multi-scale in time (Wanner and Hairer 1996), which makes training accurate surrogates that capture all timescales a difficult task (Hadjinicolaou and Goussis 1998; Anantharaman et al. 2021b). Most data-driven methods for generating surrogates fail on this task without the use of problem-specific assumptions, variable transformations (Qian et al. 2020; Kramer and Willcox 2019) or specialized training procedures (Kim et al. 2021; Ji et al. 2021). Prior work demonstrated that implicitly-trained continuous time frameworks, such as the Continuous-Time Echo State Networks (CTESN), are viable surrogate architectures for accurately learning such multi-scale dynamics (Anantharaman et al. 2021b; Rackauckas et al. 2021).

A second challenge is the difficulty of developing surrogates which do not require retraining when used in new scenarios. For instance, in design optimization as well as optimal control, many surrogates directly approximate the objective function (Peitz and Dellnitz 2018; Marzat and Piet-Lahanier 2012; Wang, He, and Liu 2017), which necessitates retraining whenever the choice of objective function changes. Most other surrogate architectures reproduce the behavior of a full model (Anantharaman et al. 2021b; Raissi, Perdikaris, and Karniadakis 2019; Benner, Gugercin, and Willcox 2015). The downside this presents is that if the causal connections between components in a model are even slightly changed then the entire expensive training procedure must be repeated.

To alleviate this issue, we sought to develop surrogate architectures that would be suitable as components within composable modeling frameworks. These causal frameworks such as Simulink, ASPEN, and ModelingToolkit (Karris 2006; Luyben 2013; Ma et al. 2021), are block-diagram system modelers where mechanistic models are composed together. These types of simulation environments provide pre-built model components of commonly reused structures, such as HVAC models (Tian et al. 2017), buildings (Wetter 2011; Wetter et al. 2019), circuit components (Cellier, Clauß, and Urquía 2007), and more, to allow engineers to easily re-purpose quantums of mechanistic models towards different applications. If such component mod-

els could automatically be reduced to an accelerated form, then all future applications which would have used the library components could instead use the same neural surrogate without requiring retraining. It is this composable and reusable formulation that we wish to target.

Block models in such modeling systems are ordinary differential equations which allow for arbitrary input functions. This can be written as:

$$u' = \varphi(u, p, t, f(t)) \tag{1}$$

$$y = \psi(u) \tag{2}$$

where u are the states of the component, f is the input function, and y are the output observables. In such a mechanistic system modeling context, systems are composed by defining the $f(t)$ by the output observables $y(t)$ of another block. Developing a surrogate architecture capable of being used in such a context thus boils down to developing continuous-time surrogates capable of capturing the system response of non-linear systems.

The problem of learning reduced system response models has been extensively studied in the context of control theory. A common case is where systems are linearized (Charlet, Lévine, and Marino 1989) about a certain operating point, after which linear model order reduction methods (Antoulas 2005; Benner, Gugercin, and Willcox 2015) or system identification methods (Ljung, Chen, and Mu 2020; Schoukens and Ljung 2019) are used to generate stand-ins for the full order model. Then, the literature on designing controllers for linear systems can then be drawn upon (Skogestad and Postlethwaite 2007). The limitation of this approach is that beyond a certain operating region, this reduced order form's system response may no longer be representative of the original system response. One method that has been used to capture nonlinear system response is the Volterra series (Cheng et al. 2017), and have been used to control multi degree of freedom systems (Feijoo et al. 2010) but the method requires the system to be time-invariant. In this work, we present a fully nonlinear machine learning based method to capture the system response of arbitrary nonlinear systems in a continuous-time architecture and show its ability to accurately accelerate causal models.

Nonlinear Response Continuous-Time Echo State Networks (NR-CTESN)

This section first summarizes the CTESN surrogate method and then describes the Nonlinear Response CTESN extension. CTESNs are a continuous-time analogue of Echo State Networks, which are a form of reservoir computing (Grigoryeva and Ortega 2018; Lukoševičius 2012). This surrogate consists of two parts: a reservoir Ordinary Differential Equation (ODE), which is cheap to simulate by design, and a projection operator from this reservoir to the reference system. While the reservoir ODE is designed by the user based on various considerations, the projection operator is trained using a linear least-squares solve. A typical formulation for the CTESN is the following:

$$r' = g(Ar + W_{hyb}x(p^*, t)) \tag{3}$$

$$x(p, t) = W_{out}r(t) \tag{4}$$

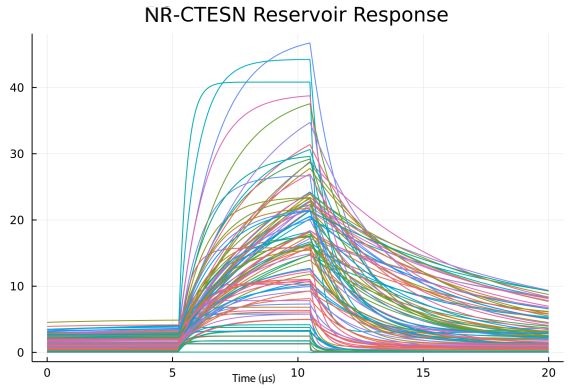

Figure 1: **Rich response dynamics of a domain-specific reservoir used in the NR-CTESN**, used to generate the surrogate of the inverter. Each line represents an output from every cell in the cellular neural network.

where Equation 3 is an example of a simple, non-parametric reservoir ordinary differential equation, written as a neural ordinary differential equation(Chen et al. 2018). A is referred to as the weight matrix in the literature (Kawai, Park, and Asada 2019), $W_{hyb}x(p^*, t)$ is a "hybrid" term used to drive the reservoir, and $x(p^*, t)$ is a candidate solution at some point in the chosen parameter space. g is an activation function, which, in addition to the weight matrix A, is chosen to control the behaviour of the full derivative term.

The CTESN is conventionally trained as follows: first, a parameter space P is chosen, which is a cross product of ranges of the system parameters. This space is then sampled, yielding $\{p_1, \ldots, p_n\} \in P$. The system is simulated at each of these parameters, yielding a training set of time series. Equation 4 refers to the second step in the training. Projections $\{W_{out}^{p_1}, \ldots, W_{out}^{p_n}\}$ are computed from the simulated reservoir r to each time series in the training set, using a QR decomposition or the singular value decomposition. Finally, an interpolating function $p \to W_{out}(p)$ is then fit. Prediction from the CTESN now follows three steps: simulation of the reservoir, constructing the projection, and then matrix-vector multiplication, as shown below.

$$\hat{x}(\hat{p}, t) = W_{out}(\hat{p})r(t) \tag{5}$$

The CTESN has been trained to produce surrogate models of Heating, Ventilation and Air Conditioning (HVAC) systems (Rackauckas et al. 2021) and quantitative systems pharmacology models (Anantharaman et al. 2021a).

There are several important considerations whilst training the CTESN. The reservoir should be always be chosen strategically. In particular, the time series from this reservoir should match key characteristics of the output time series, while being cheap to simulate. The above basic formulation may not be amenable to train systems that exhibit complex behaviour. For instance, this reservoir is largely continuous, and may not accurately capture discontinuities. If the output time series contains a discontinuity at a point in time, the reservoir should also contain it. In the next section, we shall examine a domain specific choice of reservoir which can accurately handle semi-discontinuous components seen in circuit simulations.

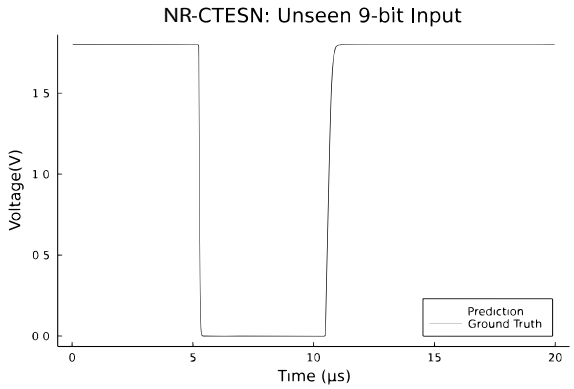

Figure 2: **Prediction of surrogate to unseen test input sequence**. The red line refers to the ground truth output from the inverter and blue line is the prediction. The surrogate is able to predict the transition from high to low at the correct times and is able to match the reference output.

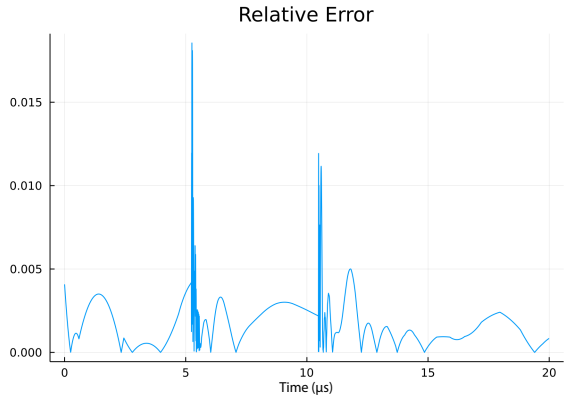

Figure 3: **Relative error of surrogate prediction at test input sequence.** The error is highest at the points of discontinuity: the places where the square signal transition from 0 to 1 and vice versa.

Training with External Forcing

Now we present a variant of the CTESN formulation that incorporates external forcing, which can then be used to capture system response to external forcing or internal disturbance. This training procedure not only trains on a bounded parameter space P like the conventional CTESN, but also on a space of external forcing functions.

Consider a bounded space of forcing functions F, such as a Fourier series or polynomials with a finite number of terms and a predefined range of coefficients. In other words, we mean to consider a bounded space of coefficients that describe a bounded space of functions. Let forcing functions $\{f_1, \ldots, f_n\}$ be sampled from this space. To incorporate forcing functions, Equation 3 may thus be modified as follows:

$$r' = g(Ar + W_{hyb}x(p^*, t) + W_f f(t)) \tag{6}$$

where W_f is constant and $f(t)$ is the additional forcing term. The reservoir r from Equation 6 is excited using each forcing function, resulting in a collection of reservoir response time series $\{r_1, \ldots r_n\}$. We also excite the original system with the same forcing functions, resulting in the time series collection $\{d_1, \ldots d_n\}$. We can now solve the least squares problem

$$\arg\min_{W_{out}} \sum_i (W_{out} r_i - d_i)^2$$

To solve this numerically, we can concatenate all the system responses like in Equation 7, and solve the following directly with a QR decomposition or the singular value decomposition:

$$W_{out}[r_1|r_2|\ldots|r_n] = [d_1|d_2|\ldots|d_n] \tag{7}$$

We dub the resultant surrogate the NR-CTESN. This idea composes naturally with the conventional CTESN training to learn a parametric surrogate. Namely, we can sample several sets of parameters at pre-defined parameter space P, compute several projections from the reservoir system responses to the original system responses, and then learn the interpolating object $W_{out}(p)$. This procedure is summarized in Algorithm 1. We also note that system response to internal disturbances and events can be learnt in the same way. Once the event in question is parametrized, a space of event parameters can be bounded and sampled, after which a projection can be fit in exactly the same fashion as above.

Algorithm 1: Training the Nonlinear Response CTESN

Input: parameter space P, space of forcing functions F, pre-designed reservoir r
Output:

1: Sample $\{p_1, \ldots p_n\} \in P$, $\{f_1, \ldots f_n\} \in F$
2: **for** p in $\{p_1, \ldots p_n\}$ **do**
3: Compute reservoir responses $\{r_1, \ldots r_n\}$ and system responses $\{d_1, \ldots d_n\}$ using $\{f_1, \ldots f_n\}$.
4: Fit projection matrix W_{out} at parameter p using Eq. 7
5: **end for**
6: Fit interpolator $\psi : p \to W_{out}$
7: **return** ψ, r

Circuit Simulation Applications: Inverter and Digital to Analog Converter (DAC)

In this section, we shall discuss how the Nonlinear Response CTESN is used to generate a surrogate of an inverter circuit. The inverter is designed using two Berkeley Short-channel IGFET Mode-4 (BSIM4) transistor models (Dunga et al. 2006) in CMOS inverter configuration. The input to this model is a 9-bit digital input with a fixed bit-width and the output is the flipped bitstream.

The first step in training a NR-CTESN is to design the reservoir. In general, the reservoir should exhibit a rich set of dynamics when excited by the digital inputs. The inverter is a mixed-signal circuit (Gielen and Rutenbar 2000), and can output both analog and digital signals depending on its physical parameters. The reservoir must also follow this behaviour.

This was achieved by use of a domain-specific reservoir in the form of a Cellular Neural Network (Chua and Yang 1988a), which consists of a grid of cells. Each cell is a

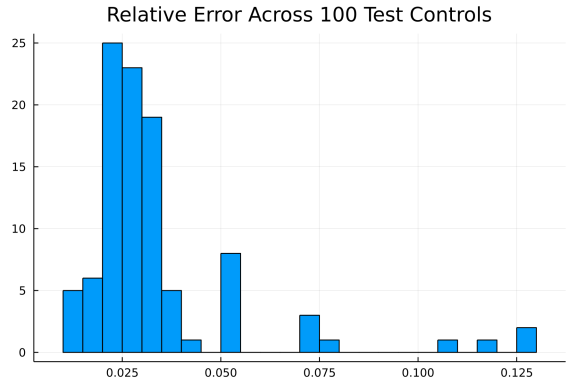

Figure 4: **Histogram of test errors**. The X axis denotes the relative error and Y axis denotes the number of test samples with the same prediction error. The majority of relative errors for the test dataset is less than 0.05.

circuit with an RC element, with additional controlled current sources denoting coupling with its neighbors. The state equations are as follows:

$$Cv'_{ij} = -\frac{1}{R_x}v_{ij} + \sum_{C(k,l)\in N_r(i,j)} A(i,j;k,l)v_{ykl} + \sum_{C(k,l)\in N_r(i,j)} B(i,j;k,l)v_{ukl} \quad (8)$$

$$v_{yij} = g(v_{ij}) \quad (9)$$

where v_{ij} is the state of each cell $C(i,j)$, the resistances R_x and capacitances C in each cell are randomly initialized, which allows them to create a range of responses. The output from each cell is the state of the cell v_{ij} filtered by an activation function g, chosen to be $\tanh$ in this example. Each cell also interacts with its neighbors via two template variables A and B, which dictate the weighted contribution from the neighbors' outputs v_{ykl} and inputs v_{ukl} respectively (Chua and Yang 1988b; Hunt et al. 1992). The size of the neighborhood $N_r(i,j)$ controls the size of the two template variables. This coupling in the system produces a rich response shown in Figure 1. For more details on the implementation, the reader is referred to the basic cellular neural network formulation in (Chua and Yang 1988b).

Since the input is 9-bit digital, there exist only $2^9 = 512$ possible discontinuous input signals to this system. The location of the discontinuity is controlled by inverter's system parameters, and the surrogate is trained on multiple sets of system parameters sampled using Latin Hypercube sampling. Additionally, it is trained with 100 inputs sampled from the possible 512. A 10×10 cellular network is randomly initialized and excited with the training inputs, and a neighborhood of 3×3 was chosen. The simulation time under consideration for training is 20 micro-seconds. A least squares projection is then calculated between the reservoir system responses and the system responses of the original system. Figure 2 shows a prediction plot of the surrogate at a test input while Figure 3 shows the relative error, which is less than 2% over the whole time series. Additionally, a histogram of relative errors was plotted in Figure 4.

On the inverter example of 44 equations a 9x acceleration of the simulation was achieved by the NR-CTESN. However, as demonstrated in previous work the CTESN architecture's acceleration increases as the size of the approximated system increases (Anantharaman et al. 2021b,a). To demonstrate this, we trained a surrogate using the same architecture on a Sky130 Digital to Analog Converter (DAC) simulated by Ngspice (Nenzi and Vogt 2011). This 1,200 equation system saw similar accuracy results but achieved a 274x acceleration, changing the simulation time from 7.3 hours to 16 minutes.

Discussion & Conclusion

This paper presents a method to learn the response of arbitrary systems to external forcing or internal events. The method presented is generalizable in that it is data-driven and places no constraints on the nature of the system being approximated. Science-guided or physics-informed priors are shown to be incorporated by the choice of reservoir. Yet, the technique is, in general, agnostic to the system being learned. The resulting surrogate is capable of generating nonlinear model order reductions of casual modeling blocks that match the reusable component architecture. This allows the NR-CTESN to automatically accelerate models from these types of modeling frameworks. When applied to the Functional Markup Interface (FMI) standard for causal models (Blochwitz et al. 2011), a widely used binary form describing models in a way that matches Equation 1, the NR-CTESN can thus be used as an FMU-accelerator, taking in FMU binary descriptions of causal components and generating new FMU binaries which reproduce the behavior at a fraction of the cost. Given the hundreds of widely used tools throughout industry engineering which use this standard[1], this technique has the potential to make a large impact on the modeling industry.

In future work, we plan to use this surrogate in order to design controls. We anticipate that the ability of the surrogate to respond to control signals at a fraction of the computational cost can accelerate design significantly. In addition, we note that the original discrete-time echo state networks have been demonstrated to be universal adaptive filters (Grigoryeva and Ortega 2018), suggesting a potential universality for the NR-CTESN. Follow up work should investigate this property or suggest variants to the NR-CTESN to correct for this ability. Finally, we note that this technique requires knowing the input states of the system and thus cannot capture the fully acasual modeling space of many system modelers such as Modelica (Mattsson, Elmqvist, and Otter 1998), Dymola (Brück et al. 2002), or SimScape (Miller and Wendlandt 2010). Extensions to the technique which cover the acausal space would expand its applicability.

[1] https://fmi-standard.org/tools

Acknowledgments

The information, data, or work presented herein was funded in part by the Advanced Research Projects Agency-Energy (ARPA-E), U.S. Department of Energy, under Award Numbers DE-AR0001222 and DE-AR0001211, and NSF awards OAC-1835443 and IIP-1938400, DARPA Agreement No HR00112190048. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Government.

References

Anantharaman, R.; Abdelrehim, A.; Jain, A.; Pal, A.; Sharp, D.; Utkarsh, U.; and Rackauckas, C. V. 2021a. Stably Accelerating Stiff Quantitative Systems Pharmacology Models: Continuous-Time Echo State Networks as Implicit Machine Learning. *bioRxiv*.

Anantharaman, R.; Ma, Y.; Gowda, S.; Laughman, C.; Shah, V.; Edelman, A.; and Rackauckas, C. 2021b. Accelerating simulation of stiff nonlinear systems using continuous-time echo state networks. *Proceedings of AAAI Symposium on Machine Learning in the Physical Sciences*.

Antoulas, A. C. 2005. *Approximation of large-scale dynamical systems*. SIAM.

Benner, P.; Gugercin, S.; and Willcox, K. 2015. A survey of projection-based model reduction methods for parametric dynamical systems. *SIAM review*, 57(4): 483–531.

Blochwitz, T.; Otter, M.; Arnold, M.; Bausch, C.; Clauß, C.; Elmqvist, H.; Junghanns, A.; Mauss, J.; Monteiro, M.; Neidhold, T.; et al. 2011. The functional mockup interface for tool independent exchange of simulation models. In *Proceedings of the 8th International Modelica Conference*, 105–114. Linköping University Press.

Brück, D.; Elmqvist, H.; Mattsson, S. E.; and Olsson, H. 2002. Dymola for multi-engineering modeling and simulation. In *Proceedings of modelica*, volume 2002. Citeseer.

Cellier, F. E.; Clauß, C.; and Urquía, A. 2007. Electronic circuit modeling and simulation in Modelica. In *Proceedings of the Sixth Eurosim Congress on Modelling and Simulation*, 1–10.

Charlet, B.; Lévine, J.; and Marino, R. 1989. On dynamic feedback linearization. *Systems & Control Letters*, 13(2): 143–151.

Chatterjee, T.; Chakraborty, S.; and Chowdhury, R. 2019. A critical review of surrogate assisted robust design optimization. *Archives of Computational Methods in Engineering*, 26(1): 245–274.

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

Cheng, C.; Peng, Z.; Zhang, W.; and Meng, G. 2017. Volterra-series-based nonlinear system modeling and its engineering applications: A state-of-the-art review. *Mechanical Systems and Signal Processing*, 87: 340–364.

Chua, L. O.; and Yang, L. 1988a. Cellular neural networks: Applications. *IEEE Transactions on circuits and systems*, 35(10): 1273–1290.

Chua, L. O.; and Yang, L. 1988b. Cellular neural networks: Theory. *IEEE Transactions on circuits and systems*, 35(10): 1257–1272.

Du, T.-S.; Ke, X.-T.; Liao, J.-G.; and Shen, Y.-J. 2018. DSLC-FOA: improved fruit fly optimization algorithm for application to structural engineering design optimization problems. *Applied Mathematical Modelling*, 55: 314–339.

Dunga, M. V.; Xi, X.; He, J.; Liu, W.; Cao, K. M.; Jin, X.; Ou, J. J.; Chan, M.; Niknejad, A. M.; and Hu, C. 2006. Bsim4. 6.0 mosfet model. *University of California, Berkeley*.

Feijoo, J. V.; Worden, K.; Garcia, P. M.; Rivera, L. L.; Rodriguez, N. J.; and Perez, A. P. 2010. Analysis of MDOF nonlinear systems using associated linear equations. *Mechanical systems and signal processing*, 24(8): 2824–2843.

Gielen, G. G.; and Rutenbar, R. A. 2000. Computer-aided design of analog and mixed-signal integrated circuits. *Proceedings of the IEEE*, 88(12): 1825–1854.

Grigoryeva, L.; and Ortega, J.-P. 2018. Echo state networks are universal. *Neural Networks*, 108: 495–508.

Hadjinicolaou, M.; and Goussis, D. A. 1998. Asymptotic solution of stiff PDEs with the CSP method: the reaction diffusion equation. *SIAM Journal on Scientific Computing*, 20(3): 781–810.

Han, Z.-H.; Zhang, Y.; Song, C.-X.; and Zhang, K.-S. 2017. Weighted gradient-enhanced kriging for high-dimensional surrogate modeling and design optimization. *Aiaa Journal*, 55(12): 4330–4346.

Hunt, K. J.; Sbarbaro, D.; Żbikowski, R.; and Gawthrop, P. J. 1992. Neural networks for control systems—a survey. *Automatica*, 28(6): 1083–1112.

Ji, W.; Qiu, W.; Shi, Z.; Pan, S.; and Deng, S. 2021. Stiff-pinn: Physics-informed neural network for stiff chemical kinetics. *The Journal of Physical Chemistry A*, 125(36): 8098–8106.

Karris, S. T. 2006. *Introduction to Simulink with engineering applications*. Orchard Publications.

Kawai, Y.; Park, J.; and Asada, M. 2019. A small-world topology enhances the echo state property and signal propagation in reservoir computing. *Neural Networks*, 112: 15–23.

Kim, S.; Ji, W.; Deng, S.; Ma, Y.; and Rackauckas, C. 2021. Stiff neural ordinary differential equations. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 31(9): 093122.

Kramer, B.; and Willcox, K. E. 2019. Nonlinear model order reduction via lifting transformations and proper orthogonal decomposition. *AIAA Journal*, 57(6): 2297–2307.

Ljung, L.; Chen, T.; and Mu, B. 2020. A shift in paradigm for system identification. *International Journal of Control*, 93(2): 173–180.

Lukoševičius, M. 2012. A practical guide to applying echo state networks. In *Neural networks: Tricks of the trade*, 659–686. Springer.

Luyben, W. L. 2013. *Distillation design and control using Aspen simulation*. John Wiley & Sons.

Ma, Y.; Gowda, S.; Anantharaman, R.; Laughman, C.; Shah, V.; and Rackauckas, C. 2021. Modelingtoolkit: A composable graph transformation system for equation-based modeling. *arXiv preprint arXiv:2103.05244*.

Marzat, J.; and Piet-Lahanier, H. 2012. Design of nonlinear MPC by Kriging-based optimization. *IFAC Proceedings Volumes*, 45(16): 1490–1495.

Mattsson, S. E.; Elmqvist, H.; and Otter, M. 1998. Physical system modeling with Modelica. *Control Engineering Practice*, 6(4): 501–510.

Miller, S.; and Wendlandt, J. 2010. Real-time simulation of physical systems using simscape. *MATLAB News and Notes*, 1–13.

Nenzi, P.; and Vogt, H. 2011. Ngspice Users Manual Version 23.

Peitz, S.; and Dellnitz, M. 2018. A survey of recent trends in multiobjective optimal control—surrogate models, feedback control and objective reduction. *Mathematical and Computational Applications*, 23(2): 30.

Qian, E.; Kramer, B.; Peherstorfer, B.; and Willcox, K. 2020. Lift & learn: Physics-informed machine learning for large-scale nonlinear dynamical systems. *Physica D: Nonlinear Phenomena*, 406: 132401.

Rackauckas, C.; Anantharaman, R.; Edelman, A.; Gowda, S.; Gwozdz, M.; Jain, A.; Laughman, C.; Ma, Y.; Martinuzzi, F.; Pal, A.; et al. 2021. Composing Modeling and Simulation with Machine Learning in Julia. *arXiv preprint arXiv:2105.05946*.

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

Schoukens, J.; and Ljung, L. 2019. Nonlinear system identification: A user-oriented road map. *IEEE Control Systems Magazine*, 39(6): 28–99.

Skogestad, S.; and Postlethwaite, I. 2007. *Multivariable feedback control: analysis and design*, volume 2. Citeseer.

Stern, R. E.; Song, J.; and Work, D. B. 2017. Accelerated Monte Carlo system reliability analysis through machine-learning-based surrogate models of network connectivity. *Reliability Engineering & System Safety*, 164: 1–9.

Tian, W.; Sevilla, T. A.; Zuo, W.; and Sohn, M. D. 2017. Coupling fast fluid dynamics and multizone airflow models in Modelica Buildings library to simulate the dynamics of HVAC systems. *Building and Environment*, 122: 269–286.

Tripathy, R. K.; and Bilionis, I. 2018. Deep UQ: Learning deep neural network surrogate models for high dimensional uncertainty quantification. *Journal of computational physics*, 375: 565–588.

Wang, D.; He, H.; and Liu, D. 2017. Adaptive critic nonlinear robust control: A survey. *IEEE transactions on cybernetics*, 47(10): 3429–3451.

Wanner, G.; and Hairer, E. 1996. *Solving ordinary differential equations II*, volume 375. Springer Berlin Heidelberg.

Wetter, M. 2011. A view on future building system modeling and simulation. Technical report, Lawrence Berkeley National Lab.(LBNL), Berkeley, CA (United States).

Wetter, M.; Coffey, B.; Haves, P.; et al. 2019. Modelica Buildings Library. Technical report, Lawrence Berkeley National Lab.(LBNL), Berkeley, CA (United States).

White, D. A.; Arrighi, W. J.; Kudo, J.; and Watts, S. E. 2019. Multiscale topology optimization using neural network surrogate models. *Computer Methods in Applied Mechanics and Engineering*, 346: 1118–1135.

Willard, J.; Jia, X.; Xu, S.; Steinbach, M.; and Kumar, V. 2020. Integrating physics-based modeling with machine learning: A survey. *arXiv preprint arXiv:2003.04919*, 1(1): 1–34.

Yildiz, A. R.; Abderazek, H.; and Mirjalili, S. 2020. A comparative study of recent non-traditional methods for mechanical design optimization. *Archives of Computational Methods in Engineering*, 27(4): 1031–1048.