Perspectives on the Integration between First-Principles and Data-Driven Modeling

Authors:

William Bradley, Jinhyeun Kim, Zachary Kilwein, Logan Blakely, Michael Eydenberg, Jordan Jalvin, Carl Laird, Fani Boukouvala

Date Submitted: 2021-11-07

Keywords: hybrid modeling, Machine Learning, neural networks, gaussian process regression, physics-informed machine learning, model calibration

Abstract:

Efficiently embedding and/or integrating mechanistic information within data-driven models is essentially the only approach to simultaneously take advantage of both engineering principles and data-science. The opportunity for hybridization occurs in many scenarios, such as the development of a faster model of an accurate high-fidelity computer model; the correction of a mechanistic model that does not fully-capture the physical phenomena of the system; or the integration of a data-driven component approximating an unknown correlation within a mechanistic model. At the same time, different techniques have been proposed and applied in different literatures to achieve this hybridization, such as hybrid modeling, physics-informed Machine Learning (ML) and model calibration. In this paper we review the methods, challenges, applications and algorithms of these three research areas and discuss them in the context of the different hybridization scenarios. Moreover, we provide a comprehensive comparison of the different techniques with respect to their differences and similarities, as well as advantages and limitations and future perspectives. Finally, we apply and illustrate hybrid modeling, physics-informed ML and model calibration via a chemical reactor case study.

Record Type: Preprint

Submitted To: LAPSE (Living Archive for Process Systems Engineering)

Citation (overall record, always the latest version):	LAPSE:2021.0801
Citation (this specific file, latest version):	LAPSE:2021.0801-1
Citation (this specific file, this version):	LAPSE:2021.0801-1v1

Perspectives on the Integration between First-Principles and Data-Driven Modeling

William Bradley[a†], Jinhyeun Kim[a†], Zachary Kilwein[a†], Logan Blakely[b], Michael Eydenberg[b], Jordan Jalvin[b], Carl Laird[b], Fani Boukouvala[a*]

[a]School of Chemical & Biomolecular Eng., Georgia Institute of Technology, Atlanta, GA, USA 30332
[b]Sandia National Laboratories, Albuquerque, NM 87185

†These authors contributed equally to this work.
* Corresponding author.
Email: fani.boukouvala@chbe.gatech.edu (F. Boukouvala)

Abstract

Efficiently embedding and/or integrating mechanistic information within data-driven models is essentially the only approach to simultaneously take advantage of both engineering principles and data-science. The opportunity for hybridization occurs in many scenarios, such as the development of a faster model of an accurate high-fidelity computer model; the correction of a mechanistic model that does not fully-capture the physical phenomena of the system; or the integration of a data-driven component approximating an unknown correlation within a mechanistic model. At the same time, different techniques have been proposed and applied in different literatures to achieve this hybridization, such as hybrid modeling, physics-informed Machine Learning (ML) and model calibration. In this paper we review the methods, challenges, applications and algorithms of these three research areas and discuss them in the context of the different hybridization scenarios. Moreover, we provide a comprehensive comparison of the different techniques with respect to their differences and similarities, as well as advantages and limitations and future perspectives. Finally, we apply and illustrate hybrid modeling, physics-informed ML and model calibration via a chemical reactor case study.

Keywords
Hybrid Modeling, Model Calibration, Physics-Informed Machine Learning, Dynamical Systems, Process Systems Design/Operations

1. Introduction

Recent developments in the broad field of data-science have led to a series of breakthroughs in Machine Learning (ML) (Qin and Chiang 2019; Lee, Shin, and Realff 2018) techniques. The Process Systems Engineering (PSE) community is having an important debate on the roles data science should have over first-principles, physics-based science (e.g., thermodynamics, transport phenomena, kinetics and mass balances) (Venkatasubramanian 2019). The key reasons for doubting the value of ML in chemical engineering is their black-box nature, a term generally used to acknowledge their poor extrapolating capabilities, lack of interpretability. and unbounded uncertainty in predictions that may not satisfy physical constraints. “Hybridization” holds the promise that the data-dependent models are more reliable because they learn from both data and physics (von Stosch, Oliveira, et al. 2014). The concept of Hybrid-Modeling is not new to PSE. Indeed, hybrid modeling techniques and applications have been growing in number since the early 90’s (Psichogios and Ungar 1992; Rico-Martínez et al. 1992; Thompson and Kramer 1994). Following an explosion of advances in purely black-box ML techniques, merging fist-principles knowledge and ML is becoming the next big trend, since it can increase accuracy and interpretability with less data.

When reviewing all potential ways to merge data-driven with physics-based models, a diverse set of approaches have been proposed in different literatures, ranging from feature engineering (i.e. careful selection of the inputs/outputs to a model), to simple additive models comprised of separate physical and data-driven equations, to advanced methods that embed physics within data-driven models using customized training and numerical techniques. In fact, concepts targeting the same fundamental idea are often disguised under different terminology, in different literatures and time periods.

Though techniques for linking mechanistic and data-driven tools are as numerous as the systems they model, some methods shown promise across multiple disciplines. In particular, three areas are reviewed in this paper are: (a) “Hybrid Submodeling” (HSM), (b) “Physics-Informed ML” (PI-ML), and (c) “Model calibration” (MC). Specifically, HM contributions focus on identifying unknown or partially-known process mechanisms constrained by known first-principles equations (typically, a system of differential equations representing conservation balances)(von Stosch, Oliveira, et al. 2014). PI-ML is in many ways a synonym to HSM, but typically refers to techniques for constrained training of deep ML models (e.g., adding physics-based loss terms, or changing weight parameters or the ML model structure) based on prior knowledge (Chen, Rubanova, et al. 2018; Raissi, Perdikaris, and Karniadakis 2019). PI-ML and HSM techniques aim to maintain the advantages inherent to data driven models (low computational cost), while making models more generalizable and physically consistent. MC is the process of updating the parameters of physics-based models with statistical techniques (typically

Bayesian methods) to compensate for the model-data discrepancy (Kennedy and O'Hagan 2001b). These terms are widely used in the PSE, ML, Statistics and Operations Research literatures.

In this perspectives paper we first review the basic principles, methods, algorithms and applications of HSM, PI-ML and MC. Most importantly, we have compiled the similarities and differences in the above areas and provide a comprehensive discussion on the challenges and limitations of each approach, potential synergies and future perspectives. Finally, we present the capabilities of each approach on a reactor modeling case study.

The sections of this paper are structured as follows. In Data-Driven Models (Section 2), we discuss purely data-driven approaches for modeling input-output data and focus on two popular techniques, namely Neural Networks and Gaussian Process Models that will be used in this work. First-Principles Models (Section 3) briefly describes the type of mechanistic or physics-based models that are amenable to hybridization and are discussed in this work. Next, in Merging Data-Driven with First-Principles Models (Section 4) we discuss potential scenarios for hybridization, as well as methods in HM, PI-ML and MC. In the Applications section (Section 5) we review areas where HM, PI-ML and MC have been applied, and in Software Implementations (Section 6) we provide a list of algorithms and software available for each of the techniques so far. Following this, this paper applies HM, PI-ML and MC concepts on a case study for a chemical reactor (Section 7), ending with a Perspectives discussion and Conclusions (Sections 8 and 9, respectively).

2. Data-Driven Models

Purely data-driven models differ from first principles models in that their parameters are fitted based on available data and often their parameters do not have a physically interpretable meaning. Thus, data-driven models are often referred to as black-box or empirical models and are primarily used for extracting correlations in data. The data may come from designed physical experiments, historical databases, or designed samples from mechanistic models or simulations, or any combination of the above sources. One popular framework involving a data-driven model is to replace a mechanistic model with a data-driven surrogate (or emulator or metamodel). This surrogate model is trained using data simulated from the mechanistic model. Surrogate modeling is primarily motivated by the cheaper computation time of the surrogate model, which can be used to accelerate time-sensitive tasks such as optimization, monitoring or control. Multiple reviews have been written on this topic within chemical engineering alone (McBride and Sundmacher 2019; Bhosekar and Ierapetritou 2018), and the reader is encouraged to consult these for specific examples. Note, however, that surrogate modeling is not considered a hybrid or physics-informed modeling technique as mechanistic information is not incorporated into the model training or

simulation. However, as done in (Schäfer et al. 2019), the surrogate model is free to be merged with other mechanistic models at time of application.

Numerous data-driven models exist and can range in complexity from generalized linear regression (e.g., linear and nonlinear terms that maintain linearity in parameters) to universal nonlinear approximators (e.g., Gaussian process, Neural Networks, Random Forests) and many more. A general distinction between "parametric" and "nonparametric" models can be found in the literature and will be adopted in this work. A data-driven model is characterized as parametric if its parameters are associated with a term have some type of physical interpretation. For example, a generalized linear regression model with linear and quadratic terms can be considered as a parametric model since the parameters associated with each term signify the importance of linear and quadratic effects on the output prediction. On the other hand, parameters in nonparametric models are not associated with terms that have any physical meaning, such as the weights and biases of the nodes of a Neural Network.

This review focuses on only a subset of nonparametric data-driven models, namely Neural Networks (NN) and Gaussian Process (GP) models, and explores how these can be merged with mechanistic knowledge. The selection of these two techniques for this review is based on the fact that they are universal approximators, which implies that they are flexible and generalizable to be merged with physics in different settings and applications. This is supported by the fact that NNs and GPs are by far the most popular data-driven models in the areas reviewed in this paper. Moreover, it can be argued that parametric models (i.e., generalized linear regression models) are in certain cases physics-informed, if *a-priori* knowledge regarding the input-output relationships is used to specify which features to include. On the other hand, the non-parametric modeling NNs and GPs are less restricted and can be modeled as black-boxes, or purely data-driven. For completeness, a brief overview of the NNs and GPs is provided below.

2.1. Gaussian Process Models

Gaussian Process (GP) modeling is a powerful tool that can be used for regression under the Bayesian framework (Rasmussen and Williams 2005). Due to its flexibility, a GP model can approximate an arbitrary continuous function (Karniadakis 2020b) and capture nonlinear dependencies between inputs (Gorbach et al. 2017). In addition, its probabilistic nature enables the incorporation of different sources of uncertainty (e.g., parameter uncertainty and experimental uncertainty) as part of the model (Higdon et al. 2004). A Gaussian Process is a flexible prior $\left(P(f)\right)$ over functions in a Bayesian updating scheme, as shown in Eq. (1):

$$P(f|Y) \propto P(Y|f) \times P(f) \quad Eq.(1)$$

Posterior ∝ Likelihood × Prior

Through Eq. (1), our prior belief on function f is updated with the given observation $Y|f$ to generate the posterior distribution $f|Y$. Gaussian Process Regression (GPR) models the relationship (f) between the inputs $X = \{\boldsymbol{x}^{(i)}\}_{i=1}^{N}$ and the outputs $Y = \left(y^{(1)}, y^{(2)}, \dots, y^{(N)}\right)^T$. Here, X is a collection of N sampled inputs $\boldsymbol{x}^{(i)}$ and Y is a collection of the corresponding N observed responses $y^{(i)}$ where $i = 1,2, \dots, N$. GPR models the mapping $f: X \to Y$. Each $\boldsymbol{x}^{(i)}$ is a p-dimensional vector that contains the input variables of the system, $\boldsymbol{x}^{(i)} = (x_1^{(i)}, x_2^{(i)}, \dots, x_p^{(i)})^T$, and we assume that each observation $y^{(i)}$ is scalar for simplicity. GPR is a generalization of a multivariate Gaussian distribution over functions (Rasmussen and Williams 2005), where for any finite selection of points $\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \dots, \boldsymbol{x}^{(N)}$, the observation vector $Y = \left(Y\left(\boldsymbol{x}^{(1)}\right), Y\left(\boldsymbol{x}^{(2)}\right), \dots, Y\left(\boldsymbol{x}^{(N)}\right)\right)^T = \left(y^{(1)}, y^{(2)}, \dots, y^{(N)}\right)^T$ follows the multivariate Gaussian distribution (MacKay 2003). A GP model is fully characterized by a mean $m(\boldsymbol{x})$ and a covariance function $k(\boldsymbol{x}, \boldsymbol{x}')$ and follows the following distribution:

$$f \sim N(m(\boldsymbol{x}), k(\boldsymbol{x}, \boldsymbol{x}')) \qquad Eq.(2)$$

where $m(\cdot): R^p \to R$ and , $k(\cdot,\cdot): R^p \times R^p \to R$,

$$m(\boldsymbol{x}) = \mathbb{E}\big(Y(\boldsymbol{x})\big) \qquad Eq.(3)$$

$$k(\boldsymbol{x}, \boldsymbol{x}') = cov\big(f(\boldsymbol{x}), f(\boldsymbol{x}')\big) = \mathbb{E}\big[\big(Y(\boldsymbol{x}) - m(\boldsymbol{x})\big)\big(Y(\boldsymbol{x}') - m(\boldsymbol{x}')\big)\big] \qquad Eq.(4)$$

Different types of covariance functions can be used to encode specific characteristics (e.g., smoothness) on the model. The squared exponential (SE) covariance function is most frequently used.

$$k_{SE}(\boldsymbol{x}, \boldsymbol{x}') = \tau^2 \exp\left(-\frac{1}{2}\sum_{q=1}^{p} w_q (x_q - x_q')^2\right) \qquad Eq.(5)$$

In Eq. 5, τ^2 and w_q are GP hyperparameters, which explain the variance of the kernel and the reciprocal of the characteristic length scale, respectively. By definition, we can construct a prior with multivariate Gaussian between the observation vector Y and the unknown function values f.

$$\begin{bmatrix} Y \\ f \end{bmatrix} \sim N\left(\begin{bmatrix} m_Y \\ m_f \end{bmatrix}, \begin{bmatrix} \Sigma & \Sigma_* \\ \Sigma_*^T & \Sigma_{**} \end{bmatrix}\right) \qquad Eq.(6)$$

where Y is the known function values of the training inputs, f is the unknown function values at the test inputs $\boldsymbol{x}^*$, and Σ, Σ_*, Σ_{**} are the covariance matrices at different pairs of training and test points. Σ is the covariance matrix between all observed points, Σ_* is the covariance matrix between the observed points

and the test inputs, Σ_{**} is the covariance matrix between the test inputs. We get the conditional distribution of f given Y by conditioning the joint Gaussian prior distribution on the observed data points (Y) (Rasmussen and Williams 2005),

$$f|Y \sim N\left(m_f + \Sigma_*^T \Sigma^{-1}(Y - m_Y), \Sigma_{**} - \Sigma_*^T \Sigma^{-1} \Sigma_*\right) \quad Eq.(7)$$

The hyperparameters θ in GPR are estimated by maximizing log marginal likelihood function (Eq. 8).

$$\max_{\theta} logP(y|X,\theta) = -\frac{1}{2} log|\Sigma| - \frac{1}{2}(Y - m_Y)^T \Sigma^{-1}(Y - m_Y) - \frac{N}{2} log 2\pi \quad Eq.(8)$$

If the data is noisy, we can replace Σ with $\Sigma + \sigma_n^2 \mathrm{I}$ where σ_n denotes the expected standard deviation of noise.

2.2. Neural Networks

Artificial neural networks (NN) (Krogh 2008) are non-parametric regressors that have received significant attention as computational resources have grown. These models have excelled in Big-Data applications such as natural language processing, imaging, and automation (Goldberg 2017; McCann, Jin, and Unser 2017). Due to their feature of being universal approximators (Cybenko and systems 1989), NNs are capable of modelling complex, nonlinear relationships in high dimensional spaces, provided that there is a deterministic relationship between inputs and outputs and sufficient/representative training data. The mathematical foundation of a NN is based on the multi-layer perceptron model (Grossberg 1988). Figure 1 provides a visual representation.

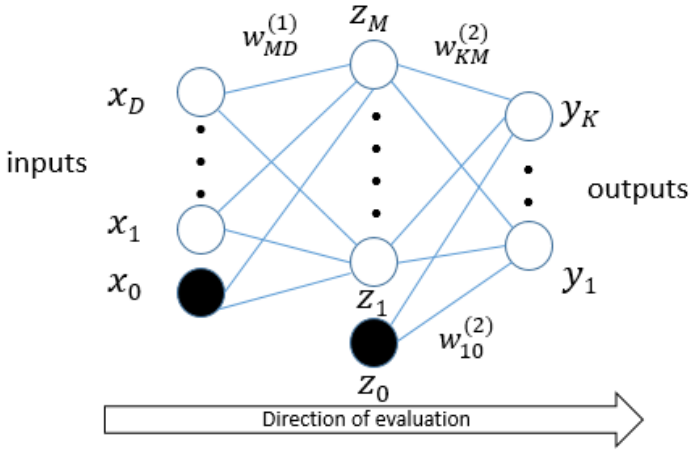

Figure 1. Neural Network structure showing input, hidden, and output variables represented by nodes. Weights are represented by connecting lines and biases by dark nodes (x_0,z_0) (Adapted from Bishop-Pattern Recognition and Machine Learning(Bishop 2006))

Here a simple two-layer model is shown that can be used to connect a set of inputs to outputs. Input nodes feed the hidden nodes as a sum of bias parameters z_o and i=1, …, D inputs x_D multiplied by j=1, …, M, corresponding weights w_{ij} (represented as lines). The resulting value at each hidden node passes through an activation function (h(x)) before feeding the output layer, which again has its own activation function (σ(x)), and weight and bias parameters corresponding to the number of outputs, k=1, …, K. Thus, the NN output can be mathematically derived as a function of input and network parameters, as shown in Eq. (9), for a 2 layer NN.

$$y_k(x,w) = \sigma\left(\sum_{j=1}^{M} w_{kj}^{(2)}\, h\left(\sum_{i=1}^{D} w_{ji}^{(1)} x_i + z_{j0}^{(1)}\right) + z_{k0}^{(2)}\right) \qquad \text{Eq. (9)}$$

The choice of the activation function can significantly affect the NN's predictive performance. Common activation functions include sigmoids, such as the logistic function, softmax function or hyperbolic tangent. These are continuous functions that mimic the behavior of biological neurons that turn on and off. In large-scale deep NNs, the piecewise linear ReLU activation function has been used widely to speed up training of network parameters, since sigmoid functions can present vanishing gradient problems (Hochreiter 1998). Eq. (9), which shows the explicit form of a NN predictor can be trained by minimizing an objective function that captures the error between predictions and data (loss function). A single output loss function is shown below, where y_n is the model prediction, $\tilde{y}_n$ is the true value with *N* total data points.

$$MSE = f(w) = \frac{1}{N}\sum_{n=1}^{N}(y_n - \tilde{y}_n)^2 \quad \text{Eq. (10)}$$

The mean-squared error (MSE) is commonly used for regression applications, where the goal is to minimize error between model prediction and the true value in the dataset. Cross entropy loss functions are often used for classification models with discrete outputs. Using the chain rule, the gradients of the objective function with respect to model weights are evaluated ($\nabla f(w)$). Weights are most commonly updated using the gradient descent Eq. 11 (alternatives include genetic algorithms and quasi-Newton methods).

$$w^{\tau+1} = w^{\tau} - \eta \nabla f(w^{\tau}) \qquad \text{Eq. (11)}$$

Hyperparameter η denotes the learning rate, which controls the step size in the optimization routine. NN evaluation, gradient calculation, and parameter updating continues iteratively until a finite set of training iterations has been reached or an objective function tolerance is satisfied.

3. First-Principles Models

First-principles models in this paper refer to models derived from fundamental laws of physics, chemistry, thermodynamics, kinetics and transport phenomena, such as mass balance and energy balances. The terms “mechanistic”, “engineering”, and “physics-based” models are used in the literature to describe such models and are used interchangeably in this paper. These models come in different forms, such as fundamental algebraic equations, or a system of Ordinary or Partial Differential Equations (ODE or PDE), or a combination of both to form a general nonlinear algebraic partial differential system of equations (NAPDE). Depending on the level of the modeling detail, or fidelity, a first-principles model can be in the form of a large complex computer simulation, with embedded hundreds of different equations and numerical techniques coupled to produce outputs, such as a Finite Element Method (FEM) model, a Computational Fluid Dynamic (CFD) simulation, or a Discrete Element Method (DEM) simulation. While first-principles models are available in different levels of fidelity and accuracy, in this work we will refer to any model derived based on fundamental knowledge as a first-principles model. Although this contribution will focus on methods for systems with continuous dynamics, there is no prohibition that such methods could not be extended to stochastic systems or systems with discrete or non-continuous relationships.

4. Merging Data-Driven with First-Principles Models

There are numerous ways to merge first-principles and data-driven tools for modeling, which span the entire gamut of applications in science and engineering where data is available and expert knowledge can be communicated in mathematical form. Broad surveys of hybrid approaches to integrate various types of theory and data can be found in (Karpatne et al. 2017; Willard et al. 2020; Rüden et al. 2019). Those surveys show that there are numerous proposed approaches which are used to inform training of data-driven models with physical knowledge, several of which are strongly application-dependent, of which only a few are covered in detail in subsequent sections. We limit our review to hybridization techniques that involve the presence of a spatio-temporal and/or algebraic mechanistic model that is used in some way during the training process (i.e., in the form of a constraint, or in combination with a data-driven component).

The main reasons for merging mechanistic knowledge with data-driven knowledge are ultimately better predictive ability of the final hybrid model (especially with reduced data requirements or in the presence of noise), and/or improved interpretability of the hybrid model. Depending on the structure of the known and unknown parts of the system, or the reason for building a hybrid model, there are multiple

structures of hybrid models found in literature. In the next section we attempt to delineate these different structures and discuss them with respect to different methodologies.

4.1. Different Structures of Combined Data-Driven and First-Principles Models

Building predictive models to find correlation or causation between system inputs and outputs is known by many names including model fitting, training, parameter estimation, statistical inference, supervised learning, or regression. Even within the subclass of regression-based problems, various methods exist to merge data and physical insights. The most prominent of these are categorized in Figure 2. Each of these schemes is reviewed in depth in this article. However, we first briefly discuss here different motives for applying these methods.

Figure 2 introduces general notation that will be used throughout this article for describing hybrid modeling formulas. In Figure 2, the output of each hybrid model Y is considered a function of system inputs x. Depending on the framework, the final predictive model may be a data-driven model $DD()$ (if first-principles is only considered during training) with parameters ϕ or a data-driven model combined with a first-principles model $f()$ with mechanistic parameters θ. The terms δ and ε are unique to the calibration framework and refer to the model discrepancy and error function, respectively.

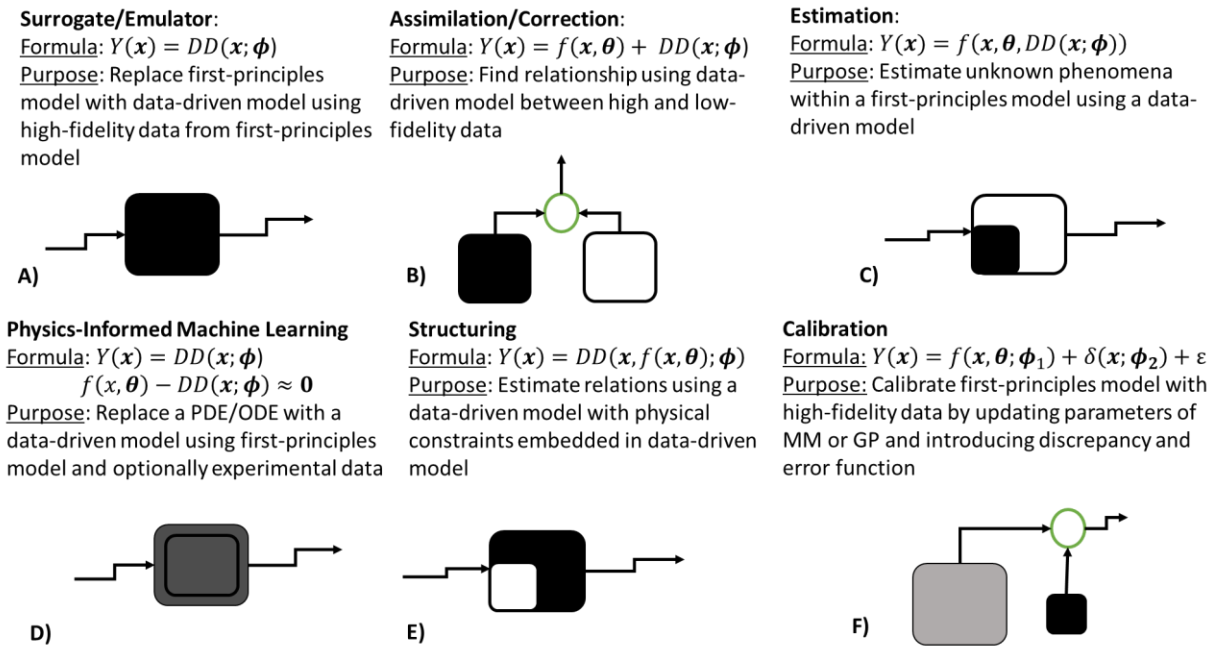

Figure 2. Structure, notation and motivation for 6 hybrid modeling scenarios addressed in this paper. θ and ϕ represent the parameters of the first principles and the data-driven model, respectively.

While no graphic could possibly capture all hybrid approaches and some approaches have no consistent literature definition, we list common hybrid approaches using Figure 2 as a guide.

- Scheme 2A: A data-driven surrogate model is used to create offline a replacement model for a mechanistic model, which can then be simulated faster online than the original mechanistic model. The surrogate model is trained only on data and no further considerations or constraints are imposed to embed physics.
- Scheme 2B: When a mechanistic model is available, but fails to accurately capture system behavior, a correction scheme may be applied by modeling the residual between low-fidelity mechanistic model and data via a data-driven model. Both the mechanistic model and data-driven correction are simulated jointly to produce the corrected output. If only the residual information is required, the HM scheme follows the scheme in Scheme 2B.
- Scheme 2C: Like the hybrid correction scheme, this approach creates a model with data-driven and mechanistic components. However, unlike the correction scheme, the output of the data-driven component aims to model a specific phenomenological relationship, yielding a more interpretable output.
- Scheme 2D: Similar to the scheme 2A in that a data-driven surrogate model replaces a more complex mechanistic model. However, PI-ML is more involved in that it accounts for mechanistic constraints during training. In recent years, of special interest is when $f(x(t), \theta)$ represents a system of differential equations and methods in derivative estimation (in particular, automatic differentiation) must be used to ensure the data-driven model is a solution of the differential form of the mechanistic model.
- Scheme 2F: The Structuring approach builds known mechanistic relationships (i.e., constraints) into an otherwise data-driven model. Although applications for this approach are more narrow than other hybrid approaches, when appropriate this scheme can improve interpretability over a purely data-driven approach while identifying both causal and correlative relationships. Depending on the embedded structure, scheme 2F may call upon similar methods as the PI-ML approach.
- Scheme 2E: Model Calibration calibrates a low-fidelity mechanistic model by utilizing both low-fidelity data (from the mechanistic equation) and the high-fidelity data (from the experiment) to generate high-fidelity output. It differs from the Correction scheme in its use to fit or 'calibrate' unknown parameters of the mechanistic model. Moreover, it attempts to more explicitly distinguish sources of error by separately modeling discrepancies due to data uncertainty and model uncertainty.

4.2. Hybrid (Sub)Modeling

Introduction

The framework most frequently associated with the keywords *hybrid modeling* in academic literature consists of constructing models that have distinct mechanistic and data-driven submodel(s). This framework is frequently referred to as hybrid semi-parametric modeling, or simply hybrid modeling (HM). To distinguish this method from other methods discussed in this review, we refer to this framework as hybrid submodeling (HSM) and the individual models merged to construct the HM as submodels (SMs). Hybrid models should not be confused with hybrid systems from control theory, which refer to control problems with both discrete and continuous decisions (Goebel, Sanfelice, and Teel 2009). HMs may consist of multiple data-driven models, mechanistic models, or combinations thereof. Methods for combining multiple data-driven models have been reviewed extensively elsewhere (Hajirahimi and Khashei 2020; Tascikaraoglu and Uzunoglu 2014; Deb et al. 2017; Zendehboudi, Rezaei, and Lohi 2018). Instead, this review focuses on HMs that merge both mechanistic and data-driven sub-models.

Methods

Applications in model building for process systems engineering (PSE) that call for HSM can generally be divided into two categories: mechanism estimation and mechanism correction. In the case of mechanism estimation, the modeler has available a mechanistic model (e.g. a series of conservation balances) within which one or more physical relationships is unknown or partially-known (e.g., a reaction rate or friction term) and estimates the unknown phenomena using a data-driven model. Alternatively, if the mechanistic model is available, but there exists a substantial discrepancy between observed data and mechanistic model predictions, the data-driven model is used to model the discrepancy (Su et al. 1992; Lee, Vanrolleghem, and Park 2005; Duarte, M. Saraiva, and Pantelides 2004; Chen et al. 2004). This latter formulation, sometimes referred to as discrepancy or residual modeling, can be viewed as a more general version of mechanism estimation and is herein referred to as mechanism (or model) correction. Naturally, formulations that combine mechanism estimation and correction have also been proposed (Thompson and Kramer 1994; Wang et al. 2010; Chen and Ierapetritou 2020). Equations 12-14 show potential formulations of each for the case where the HSMs can be modeled by differential equations.

Mechanism Estimation

$$\frac{dx_i}{dt} = f(x(t), c(t), \theta), DD(x(t), \phi) \qquad \text{Eq. (12)}$$

Mechanism Correction

$$\frac{dx_i}{dt} = f(x(t), c(t), \theta) \quad \text{Eq. (13)}$$

$$x_{\Delta t+t} = DD(x_t, \phi) \quad \text{Eq. (14)}$$

In the above equations (Eq 12-14), the state variables x are modeled by mechanistic model $f()$, which is a function of the external forcing (i.e., control or operating) variables $c(t)$, mechanistic parameters θ, and data-driven relationships parameterized by ϕ. It is worth noting that mechanistic parameters μ can be constant values or represent mechanistic relationships with parameters that must also be estimated. When using formulas of the form of Equation (12), the mechanistic and data-driven relationships must be evaluated simultaneously as the differential equations are integrated. Conversely, in the mechanism correction framework (Eq. 13-14) the mechanistic SM can be simulated independently of the data-driven SM; the outputs of the mechanistic model along with process data are used as inputs to the data-driven SM. The data-driven submodel can then predict the system state directly or predict the state residual, which are added to the mechanistic submodel predictions for the final state prediction. Schematic representations of the above equations are presented in Figure 3.

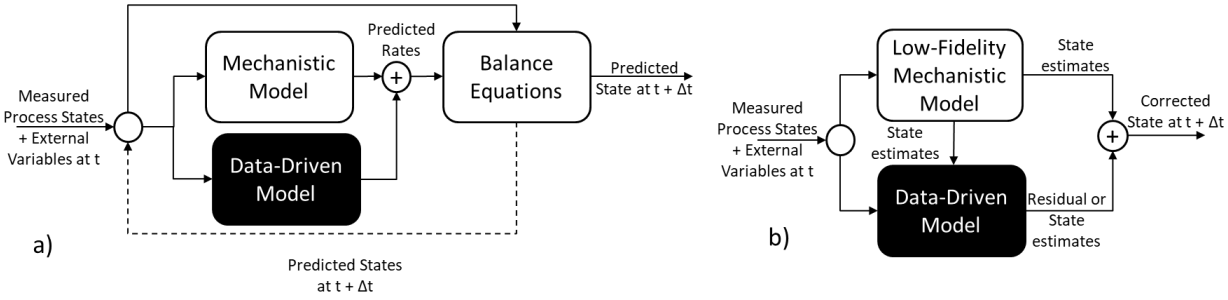

Figure 3. Information flow between submodels for HMs for a) mechanism estimation and b) mechanism correction.

While not representative of all HSM arrangement possibilities, HSM arrangements depicted in Figure 3 attempt to portray a general class of methods for arranging HSMs for modeling dynamic data. Notably, many authors choose to distinguish frameworks in Figure 3 based on whether information is exchanged between SMs "in series" or "in parallel" (von Stosch, Oliveira, et al. 2014; van Can et al. 1997). However, as can be seen from Figure 3, this can be an oversimplification since HSM for model estimation and model correction often exchange information in ways that are both serial and in parallel. This is especially true for differential equation models with mechanistic and data-driven terms. In the process of integrating the differential equations, the SMs repeatedly exchange information. To avoid possible confusion, we use mechanism estimation and correction to distinguish the end use of the HSM

framework. A study by Agarwal et al. exhaustively compared possible ways to arrange hybrid submodels for applications in modeling and control (Agarwal 1997).

Whether used for mechanism estimation or correction, building the HSMs follow the general steps of data preprocessing, model-fitting, validation, and testing/implementation. Preprocessing can be further divided into steps for outlier removal, interpolation of missing data, and feature selection. Bollas et al. demonstrated an approach for identifying significant features when constructing HSMs (Bollas et al. 2003). Preprocessing can sometimes include steps to correlate measurable observables to unmeasurable quantities of interest, using a combination of data-driven and mechanistic modeling, creating the equivalent of a soft sensor. When these correlation models are combined with an overall system model, they may be referred to as submodels of the HM (Meng et al. 2019; Thompson and Kramer 1994; Gibert, Sànchez-Marrè, and Izquierdo 2016; Lopez et al. 2020).

Challenges

An important question to ask is what role the mechanistic SM plays during training of the data-driven SM. Merging SMs during parameter estimation invariably increases the modeling effort and compute time. The answer generally depends on whether the features of the data-driven model can be measured directly. In the case of mechanism correction, the residual often can be computed with a pretrained mechanistic SM and measurement data, and the data-driven SM is trained separate from the mechanistic SM. However, when large gaps exist in measured data, fitting the data-driven correction model together with the mechanistic model can lead to improved performance (Wu and Movellan 2012).

Likewise, parameter estimation of the data-driven SM for mechanism estimation can be performed in the presence (Psichogios and Ungar 1992; Schubert et al. 1994) or separately from (Fiedler and Schuppert 2008) the mechanistic model relationships. However, for systems formulated by differential algebraic equations (DAEs) often the data-driven SM is used to estimate a rate term which is not directly measured. To avoid using the mechanistic SM during training of the data-driven SM, early works proposed estimating the rates from state data, using for example finite differences (van Can et al. 1997). Yet for highly nonlinear, sparse or noisy data the accuracy of these rate estimates can be inadequate. For such situations it has been shown that integration of the differential equations during training is essential for accurate modeling (van Can et al. 1999) and for enforcing physical constraints (Oliveira 2004).

When combined training of the data-driven SM with the mechanistic DAEs cannot be avoided, several strategies are available for managing computational costs. One strategy, applicable when multiple formulations of the data-driven model are being consdered, is the incremental approach. In this approach,

the modeler chooses to first fit multiple data-driven models separate from other mechanistic relationships and then selects the best-performing data-driven SM via cross-validation before training data-driven and mechanistic SMs together (Kahrs and Marquardt 2008). A similar incremental approach has been proposed when estimating the parameters of both the mechanistic and data-driven SMs (Yang, Martin, and Morris 2011) as well as selecting candidate mechanisms from a basis set via sparse regression (Willis and von Stosch 2017). Another strategy is to reduce the cost of integrating the DAEs. The integrated framework proposed by Ungar et al., relies on forward sensitivity analysis for parameter estimation, which is computationally expensive since the number of differential equations that must be integrated increases with the number of parameters (Narayanan et al. 2019). Thus, one way to reduce compute time is to leverage less precise numerical methods when finer accuracy is not needed. In (Oliveira 2004), researchers compared the computational efficiency of the Euler method with a more sophisticated Runge-Kutta-based discretization method, showing the latter required an increase in computation time by two orders of magnitude. Other works have investigated methods for reducing computation with adaptive step sizes (de Azevedo, Peres, and von Stosch 2015). Most recently, strategies for reducing compute costs by avoiding integration of the sensitivity equations altogether have been investigated. This strategy has been made possible by the recent development of software integration routines with pervasive automatic differentiation (AD) (Rackauckas et al. 2018). In brief, by avoiding the integration of the sensitivity equations, there is potential to prevent the explosion in computational cost for training DAE models with a large number of parameters (Rackauckas, Ma, et al. 2020; Chen, Rubanova, et al. 2018).

As a final consideration, it is often the case that both the mechanistic and data-driven SMs have parameters that require identifying (for examples see (Yang, Martin, and Morris 2011)). However, works that simultaneously train the mechanistic and data-driven submodels predominantly assume that the mechanistic model has no parameters or that their parameters are fixed at known values. Thus, additional research is needed to weigh the merits of schemes that estimate parameters, whether simultaneously or sequentially, of multiple SM types. While identifying mechanistic parameters offers the potential for increased interpretability, a foreseeable challenge is the presence of multiple local minima of the data-driven SM, which may make finding meaningful values of mechanistic parameters difficult.

4.3. Physics-Informed Machine Learning

Introduction

An area that has gained significant attention in recent years is physics-informed machine learning (PIML). It has emerged as a way to take advantage of major advances in machine learning for the purposes of surrogate modeling and system identification, while still enforcing physical knowledge that is known about the system at hand. This can assuage concerns that ML approaches abandon all of the useful

information given from first principles models or that data-driven models may give erratic results. While some of the proposed mathematical foundations for enforcing differential-equation physical constraints or knowledge (e.g. boundary and initial conditions) into ML models have been around for decades (Lagaris, Likas, and Fotiadis 1998), increased computational power, major break-throughs in ML, and the power of automatic differentiation (Baydin et al. 2017) has allowed for physics-informed machine learning models to show promise across many disciplines and applications. These models incorporate deep-learning architectures such as neural networks or gaussian processes, but also leverage known physical constraints specific to the application. This may be particularly useful in applications where the underlying physical models are too expensive to solve using traditional methods or where first-principles knowledge is helpful yet insufficient in predicting relationships between inputs and outputs. Major work has been done in the areas of differential equations, physics, power systems, and robotics. Many process systems engineering models rely on analogous conservation balances that can also be formulated into a physics-informed ML model.

Methods

Standard deep-learning models take labeled data and map the relationships between inputs and outputs. Labeled data refers to data points that have a measured or ground-truth output, which is very useful for training, validating and testing against an objective measure of model accuracy. Even when input-output relationships are highly-nonlinear and not completely understood, ML models have shown great success as function approximators with sufficient labeled training data. A great insight in the physics-informed ML literature is the ability to use unlabeled data points in order to penalize constraint violations over the entire input space of the ML model (Raissi, Perdikaris, and Karniadakis 2019). Unlabeled data refers to points in the input space without available measured or ground truth values to compare model predictions to. Though most ML training routines would ignore parts of the input space without data, physics-informed training structures can instead enforce general physical knowledge of the system we know to be true. While many variations and applications exist, the most common method is to add these constraint violations directly to the loss function during ML model training (see Figure 5). The loss function evaluates how well the model is performing and is minimized by changing model parameters over the training routine. By including knowledge directly in the loss function, the resulting model parameters will be biased towards the embedded knowledge. This can be thought of as a soft constraint, as there is no guarantee that it will be satisfied. Instead, the learning task balances the two learning goals simultaneously: improving model agreement with data and adjusting model parameters to follow known constraints. This is useful if the full engineering model is too computationally expensive to

simulate repeatedly. For example, say we want to fit a surrogate ML model to labeled data with inputs *x* and *t* and output *y* (Eq. 15)

$$y(\boldsymbol{t}, \boldsymbol{x}) = DD(\boldsymbol{t}, \boldsymbol{x}, \boldsymbol{\phi}) \qquad \text{Eq. (15)}$$

Furthermore, in the physical system we are modeling, we know the functional form of constraints that depends on input or output variables, such as the generic equation and inequality shown in Eqs. 16-17.

$$f(\boldsymbol{t}, \boldsymbol{x}, y) = 0 \qquad \text{Eq. (16)}$$

$$g(\boldsymbol{t}, \boldsymbol{x}, y) \leq 0 \qquad \text{Eq. (17)}$$

If these constraints apply to the full input domain, this input space can be discretized and the constraint can be numerically evaluated at the discretized points, what some in the literature refer to as collocation points. A loss function with a soft constraint can then be formulated as shown in Eq. (18).

$$Loss = \lambda_y \frac{1}{N} \sum_{i=1}^{N} \left| DD\left(t_y^i, x_y^i\right) - y^i \right|^2 + \lambda_f \frac{1}{N_f} \sum_{j=1}^{N_f} \left| f\left(t_f^j, x_f^j, y\left(t_f^j, x_f^j\right)\right) \right|^2 +$$

$$\lambda_g \frac{1}{N_g} \sum_{k=1}^{N_g} \max\left(0, g\left(t_g^k, x_g^k, y\left(t_g^k, x_g^k\right)\right)\right) \qquad \text{Eq. (18)}$$

Here, the first term evaluates the mean squared error between a set of *N* labeled data points and the ML model's prediction and would be common to the loss function in any regression task. The second and third terms evaluate the known equality and inequality constraints respectively at any unlabeled data point of sets N_f and N_g respectively. Hyperparameters (λ) can be adjusted to control the relative weight of these terms. In all deep learning models, finding optimal hyperparameters can be expensive and most methods simply consider grid search techniques to balance over and underfitting of validation data. In this way, the mechanistic terms in Eq 18 can be treated the same as regularization terms used in fully DD models.

A more rigorous variation of the soft constraint method outlined above uses Lagrangian optimization theory and the hyperparameters become Lagrangian multipliers (Fioretto et al. 2020). Depending on the final values of λ, this method can provide some level of guarantee for the constraints at the training values. It also may avoid some ill-conditioning issues in the simpler penalty approach. A full description of this Lagrangian method applied to deep learning models can be found in (Fioretto et al. 2020) and the underlying theory in (Freund 2004).

Another common way to embed physical knowledge into ML models is to pre-train, or initialize, model parameters. This could be done using a subset of data collected from a single larger system or using a separate data set from a system known to share physical characteristics with the target system. Since many of the ML algorithms depend on stochastic gradient descent, a pre-trained model can help to

avoid local minima that don't obey physical knowledge. This is especially helpful when fitting deep ML models to sparse data and is often called transfer learning. A full review of transfer learning can be found in (Pan and Yang 2010). This idea has further been used in chemical engineering literature under the term model migration (Lu, Yao, and Gao 2009), where their goal is to utilize underlying physical knowledge of an old process model to inform their new process model, then use a small data set to calibrate new process parameters and conditions using NNs as surrogates. In (Luo and Gao 2015), model migration is explored for Gaussian process models used to predict chemical reactor performance. A GP model is trained under certain concentrations and temperatures, then model migration is used to make predictions at extrapolated points. Furthermore, the authors explain an approach to perform model migration and process optimization simultaneously.

Another area of research under the umbrella of PIML has looked into designing the architecture of ML models to confer known physical knowledge (Xia, Feng, and Wang 2008). This is the Physics-constrained Structuring approach depicted in the HM comparison figure presented earlier (Figure 2, scheme E). By leveraging part of the ML model to impose physics, the ML model becomes more interpretable while guaranteeing domain knowledge is satisfied. This approach is especially relevant to NN models that are highly customizable and modular in structure. A simple example of this could include the use of ReLU or softmax activation functions in order to enforce non-negativity in output or intermediate variables. Jia and co-workers (Jia et al. 2020) uses this idea when modeling the density of water as a function of lake depth to ensure their prediction model monotonically increases in density as a function of depth, as this is a well-known fluid property. Monotonicity constraints have been systematically applied to other data driven model structures, such as decision tree models (Potharst and Feelders 2002). This algorithm has been implemented into Python packages such as LightGBM (Ke et al. 2017).

More complex physical knowledge can be built into ML architectures, such as local mass balances in differential form. In many applications, encoder-decoder structures are used to filter out measurement noise and transform data into a lower dimensional, information rich space (Vincent et al. 2008; Chen, Dai, et al. 2018; Kim, Choi, et al. 2020). Work in turbulence modeling has used non-trainable layers that compute the continuity equation to provide constraints that are built into the model itself (Mohan et al. 2020). A schematic of their approach is shown in Figure 4.

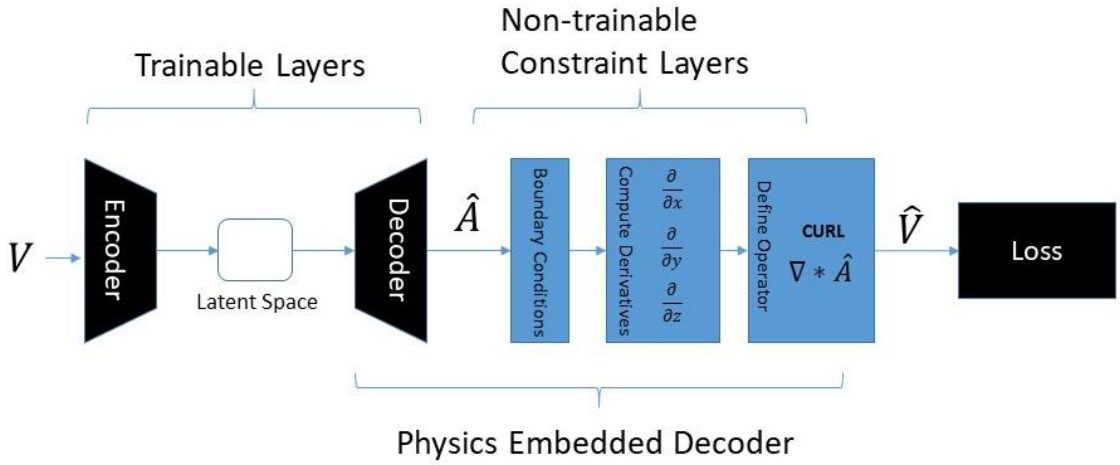

Figure 4. Example of enforcing physical knowledge within NN architectures via constraint layers and encoder-decoder layers.

In the above schematic, *V* represents input velocity data, which has some noise associated with it. $\hat{V}$ represents the "coarse-grained" representation of the velocity after passing through the autoencoder to filter out noise. $\hat{A}$ represents the vector potential which is explicitly defined in the physical fluid dynamics balance equations. Autoencoders are common when working with black-box models, since it is important to fit underlying signal instead of data-set noise. They comprise of two parts: an encoder which takes the input data and recasts into a lower dimensional representation and decoder that transforms the low dimensional data back into the same dimensional space as the encoder input. Local coherence in physical data motivates the use of Convolutional Neural Network (CNN) architectures, as high-dimensional data that has spatial dependencies on nearby features (e.g. pixels in image processing) use CNNs to abstract spatial information from data. The final Convolutional Neural Network (CNN) has a kernel defined in a way that acts as the ∇ operator. For a full discussion and explanation of CNNs see (Dhillon and Verma 2020). The overall model structure allows a lower dimensional representation to be found, while still enforcing the continuity equation. In the same field, researchers have used customized CNNs to incorporate various physical knowledge, such as uniform motion, rotation, and scaling (Wang, Walters, and Yu 2020). Their results show significant improvements in generalizability.

The area with the most extensive literature in Physics-Informed ML deals with the solution of complex differential equation models. ML models, including NNs, have long been used in the numerical solution of these systems due to their characteristic as universal function approximators (Dissanayake and Phan-Thien 1994; Cybenko and systems 1989). Many approaches to this lean heavily on the work of

(Lagaris, Likas, and Fotiadis 1998), which formulates a trial solution to a differential equation as shown below.

$$\psi_t(\boldsymbol{x}) = A(\boldsymbol{x}) + F(\boldsymbol{x}, N(\boldsymbol{x}, \boldsymbol{w})) \qquad \text{Eq. (19)}$$

In Eq. (19) $A(\boldsymbol{x})$ represents initial and boundary condition contributions to the solution, while F represents the functional form of the solution that is decoupled from these conditions and N represents the output of the neural network model. The trial solution in Eq. 19 can be analytically differentiated with respect to the independent variable (x) in order to match the original differential form and from here an unconstrained optimization problem is formulated to minimize violation of the differential equation with network parameters (w) as the decision variable. The authors demonstrate several functional forms of the trial solution for first order ODEs, second order ODEs, systems of ODEs and nonlinear PDEs. It is important that samples from an analytical solution are available and that the candidate solution can be constructed from the known differential equation. Subsequent work in this area has increased the dimensional size of problems solved through this DL method (>200 dimensions) (Sirignano and Spiliopoulos 2018) and various generalized algorithms have been developed to automatically generate trial solutions. In (Sirignano and Spiliopoulos 2018), authors propose a "Deep Galerkin Method" algorithm which generates random sample points to avoid the need for traditional mesh methods (intractable in high dimensions) and utilizes stochastic gradient descent to find parameters that minimize squared error between the model prediction and the analytical solution.

Building off the work of (Lagaris, Likas, and Fotiadis 1998), Raissi et al. utilized the power of automatic differentiation to preclude the need for analytical derivatives and applied PDE knowledge to feed-forward neural networks, which they call "physics-informed neural networks" (PINNs) (Raissi, Perdikaris, and Karniadakis 2019). The PINN formulation is analogous to the Eqs 16-18, where f becomes a differential constraint computed via automatic differentiation. They show that this is a very powerful technique for modeling PDE's when model outputs are differentiable with respect to model inputs. A schematic of the PINN is shown in Figure 5.

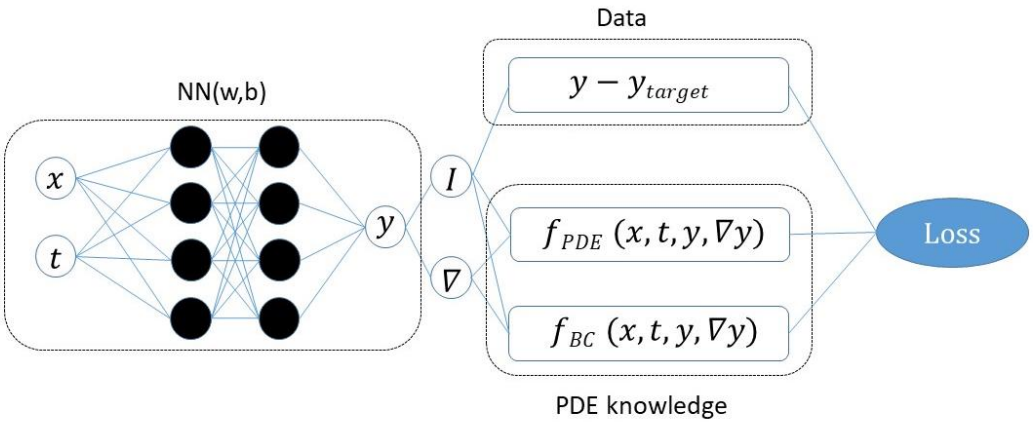

Figure 5. Schematic representation of Physics-Informed NNs Used for PDE's

The authors use sampled points across the problem domain or collocation points to calculate coherence to PDE and BC knowledge, while using analytical solutions for data. This work uses primarily Dirichlet boundary conditions. Despite the heavy focus of PI-ML methods on NN models, Gaussian Process models have also been employed. In (Raissi, Perdikaris, and Karniadakis 2017b), the authors present how numerical Gaussian Process regression models can be used to solve PDEs from noisy data. They use a backwards Euler approximation of the PDE to express each solution point as a function of previous Gaussian Process prior. Hyperparameters in the kernel model are optimized at each step with the resulting Gaussian Process model able to predict data for the next step. By linking each time step in this way, PDE knowledge is incorporated into the DL model. Their results show that for classic physics-based PDEs accurate solutions can be found with sparse training data. This method did not perform as well with non-linear operators, since linear approximations must be used. Since then, an inference procedure usually nonlinear Gaussian Processes has been developed that can be considered a Bayesian version of PINNs (Yang, Wong, and Kou 2021). One major advantage of the physics-informed Gaussian process models (PIGPs) is the estimate of uncertainty which is absent in analogous PINN models. A full review on the comparison between PINNs and PIGPs can be found in (Kevrekidis, Cuevas-Maraver, and Saxena 2020).

Challenges

Overall, the methods and applications outlined show many current strengths, along with limitations and areas for future work. Physics-informed ML show great potential in the field of PSE and can be directly applied to many already existing research directions. A direct application includes improving the robustness of PSE surrogate models which are commonly used to approximate complex unit operations with the ultimate goal of optimization or control. To be suitable for control, a model must define the

relationship between system states and controllable variables. However, PINNs designed for the solution and fitting of differential equations do not use states but rather independent variables such as time and spatial coordinates as inputs to enable derivative calculation. Recent work has sought to extend PINNs to state-space modeling for control (Arnold and King 2021), though a comparison with standard data-driven approaches is lacking. Other PIML approaches formulate the PINNs as a purely state-space model. These approaches owe their success to taking advantage of the stationary nature of the problem (Fioretto, Mak, and Van Hentenryck 2020) or system symmetry (Lutter, Ritter, and Peters 2019). An issue with these approaches is their complex integration between the physics and data-driven model may not be extensible to other systems. Assuming these barriers can be overcome, PIML may enable PSE surrogates to find optimal values that are more consistent with the ground truth physics. Model Predictive Control (MPC) is an area where hybrid model structures can thrive due to their quick computation and on-line 'learning'. Analogous to how NN models in robotics control can be embedded with Newtonian physics balances to achieve physics-informed online learning, one may imagine incorporating an energy balance into an ML based MPC model for process temperature regulation.

On the other hand, PINNs still share many of the weaknesses of traditional NNs: lack of interpretability, heavy initial training cost, challenges with extrapolation, and the requirement of many representative data or collocation points. Though they can mitigate some of the previously listed concerns, it is doubtful that they can displace many of the heavier first-principles models that have long been accepted academically and in industry. While PINNs have a great number of studies for systems well-posited physical relationships, there is sparsity of studies in areas relevant to chemical process systems, such as optimization of systems with time-varying control actions, disturbances, and poorly understood physics. Initial studies comparing PINNs with HSMs employing numerical methods have shown PINNs to be less flexible and less accurate (Mitusch, Funke, and Kuchta 2021). Another study explored the forward and inverse solution of a reaction system using a modified PINNs framework and illustrated how unmeasured state data could limit the identifiability of the fitted reaction parameters (Gusmão et al. 2020). A more general treatment of challenges associated with PINNs can be found in (Karniadakis et al. 2021).

However, there has also been some work to overcome these weaknesses for specific applications. Addressing the issue of lengthy training times, conservative PINNS (cPINNs) (Jagtap, Kharazmi, and Karniadakis 2020) and extended PINNs (XPINNs) (Karniadakis 2020a) are frameworks for sub-dividing the spatio-temporal domain into intervals regressed by separate NNs. Although not demonstrated, this approach lends itself to high parallelizability. More difficult to quantify, however, is the added time required to train hyperparameters of multiple NNs and decide how to properly decompose the spatio-

temporal domain into an appropriate number of sub-intervals. Previously discussed Langrangian Dual techniques can improve and automate hyperparameter training. Further speed-ups in training and prediction accuracy have been achieved by employing adaptive activation functions (Jagtap, Kawaguchi, and Karniadakis 2020) and modifying gradient contributions (Wang, Teng, and Perdikaris 2020).

In all these examples, the authors demonstrate that the PINN framework supersedes the accuracy of purely data-driven approaches and requires less data. They also simulate faster than and are competitive in accuracy to the analytical model, at least for most interpolation tasks and certain extrapolation tasks (discussed later in Sec. 8). Moreover, in certain circumstances the PINN can predict more accurately than the analytical model. For example, in a robotics control study the PINN was shown to outperform the control actions of an analytical model since the PINN was able to be updated online (Lutter, Ritter, and Peters 2019). Work on power systems have shown PINNs to generate far more accurate solutions than other reduced order modeling methods (Fioretto, Mak, and Van Hentenryck 2020). In many ways, physics-informed ML represents a bridge between two major areas of scientific computing which may enable tackling real-world problems requiring both theoretical and empirical resources.

4.4. Model Calibration

Introduction

In Model Calibration (MC), the typical scenario includes three components (a) a complex computer simulation that contains many equations and unknown parameters (e.g., Finite Element Model, Computational Fluid Dynamics model, etc.), (b) observed or experimental data (considered high-fidelity data), and (c) a surrogate model (typically a Gaussian Process (GP) model) that is used to calibrate the computer simulation and capture the discrepancy between the computer simulation and the observed data. Computer simulations are essential to understand and predict complex systems (Santner, Williams, and Notz 2003), however, simplifying assumptions often used in computer simulations cause a discrepancy between the computer simulation output and the observed data. Moreover, the parameters of the computer simulation are often unknown. In this scenario, a GP model can be trained with the low-fidelity simulation output and high-fidelity experimental data by adjusting the parameters (i.e., calibration parameters ($\boldsymbol{\theta}$)) of the computer simulation, and thus the calibrated model predicts the high-fidelity experimental data. This is the premise of model calibration. MC can be viewed as an example of multifidelity modeling in the sense that it considers both high-fidelity and low-fidelity data.

Methods

Variable inputs ($\boldsymbol{x}$) and the calibration parameters ($\boldsymbol{\theta}$) are two kinds of inputs that are required to run a computer model. Variable inputs ($\boldsymbol{x}$) are the inputs that can be observed or often controlled when

conducting physical experiments. Calibration parameters ($\boldsymbol{\theta}$) refer to any physical or tuning parameters that are unknown or not measurable from the experiments but are required to run the computer simulation. Both inputs are required to run the simulation, but calibration parameters ($\boldsymbol{\theta}$) are unknown to the modeler. For example, when designing the distillation column for the separation of two unknown chemicals, the number of stages or the reflux ratio are the examples of variable inputs, while the enthalpy of unknown chemicals are the examples of calibration parameters since it is left unknown to the modeler but required to run the simulation. Generally, we assume calibration parameters ($\boldsymbol{\theta}$) to be constant over the experiment, and they can either be physically interpretable (e.g., enthalpy of unknown chemicals) or have no physical meaning (e.g., tuning parameter of the surrogate model). Figure 6 describes the model calibration process.

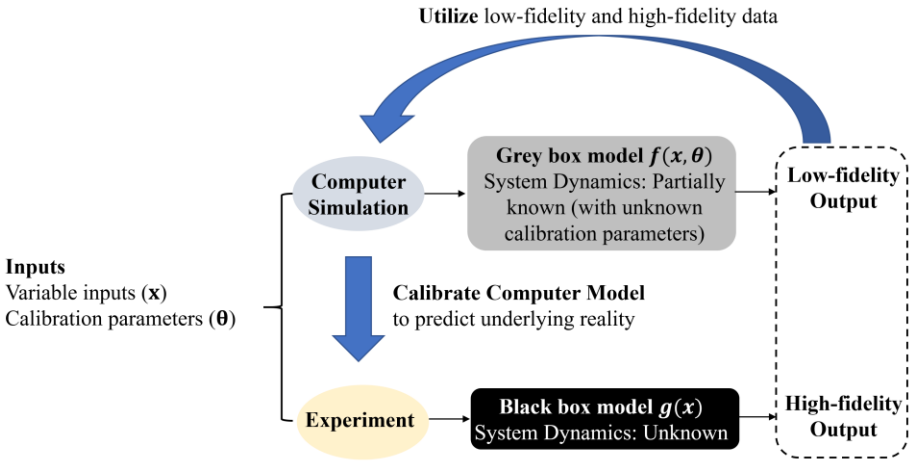

Figure 6. Schematic Illustration describing Model Calibration Process

Since the true values of the calibration parameters are unknown, there is no way to observe the computer simulation output without specifying them (Kennedy and O'Hagan 2001a). To run the simulation, researchers may try a brute force approach wherein they choose multiple values of the calibration parameters and settle for the value that minimizes the error between the computer simulation output $f(\boldsymbol{x}, \boldsymbol{\theta})$ and the real data $Y(\boldsymbol{x})$. However, this approach is challenging when the dimension of calibration parameter space ($\boldsymbol{\theta}$) becomes large, or each computer simulation run is expensive. To overcome this issue, Kennedy and O'Hagan (Kennedy and O'Hagan 2001a) proposed a combined framework to calibrate the computer model with the observed data,

$$Y(\boldsymbol{x}) = f(\boldsymbol{x}, \boldsymbol{\theta}) + \delta(\boldsymbol{x}) + \varepsilon \qquad Eq.(20)$$

where $Y(\boldsymbol{x})$ is the output of the system or real data from a physical experiment, $f(\boldsymbol{x}, \boldsymbol{\theta})$ is the computer model output, $\delta(\boldsymbol{x})$ is a discrepancy function to capture errors between the model and the data, and ε is

the measurement error which captures the effect of noise or failed-to-include variables in the system (Joseph and Yan 2015). $Eq.(20)$ is the basic framework for the model calibration process.

In model calibration, we postulate a prior distribution for the computer model, discrepancy function, and the error term, and update the distribution by conditioning the joint Gaussian prior distribution on the computer simulation and the experimental data. When we construct joint data vector $\boldsymbol{d}$ with simulation data (Y_s) and the observed data (Y), $\boldsymbol{d} = [Y_s,\ Y]^T$, the likelihood for the vector $\boldsymbol{d}$ follows the distribution in $Eq.(21)$.

$$P(\boldsymbol{d}|\boldsymbol{\theta},\boldsymbol{\phi}) \propto |\Sigma_d|^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(\boldsymbol{d}-\boldsymbol{\mu_d})^T \Sigma_d^{-1}(\boldsymbol{d}-\boldsymbol{\mu_d})\right) \qquad Eq.(21)$$

Note that we have two sets of parameters ($\boldsymbol{\theta}$: calibration parameters, $\boldsymbol{\phi}$: GP hyperparameters) in the model. The mean and variance of the likelihood for the vector $\boldsymbol{d}$ are shown in $(Eqs. 22-23)$.

$$E(\boldsymbol{d}|\boldsymbol{\theta},\boldsymbol{\phi}) = \mu_d = \begin{bmatrix} m_{Y_s}(\boldsymbol{x}) \\ m_{Y_s}(\boldsymbol{x},\boldsymbol{\theta}) \end{bmatrix} + \begin{bmatrix} 0 \\ m_\delta(\boldsymbol{x}) \end{bmatrix} \qquad Eq.(22)$$

$$var(\boldsymbol{d}|\boldsymbol{\theta},\boldsymbol{\phi}) = \Sigma_d = \begin{bmatrix} \Sigma_{Y_s,Y_s} & \Sigma_{Y_s,Y} \\ \Sigma_{Y,Y_s} & \Sigma_{Y,Y} \end{bmatrix} = \begin{bmatrix} \Sigma_{Y_s,Y_s} & \Sigma_{Y_s,Y} \\ \Sigma_{Y,Y_s} & \Sigma_y + \Sigma_\delta + \Sigma_\varepsilon \end{bmatrix} \qquad Eq.(23)$$

where $m_{Y_s}(\cdot)$ and $m_\delta(\cdot)$ denotes the mean vector of Y_s and δ, and Σ_{Y_i} and Σ_{Y_i,Y_j} are the variance matrix of Y_i and covariance matrix between Y_i and Y_j, respectively. Note that the calibration parameter $\boldsymbol{\theta}$ and the discrepancy function δ are introduced to model the observed data Y.

We can use Bayesian inference to get the posterior distribution of the GP hyperparameters $\boldsymbol{\phi}$ and the calibration parameters $\boldsymbol{\theta}$: $P(\boldsymbol{\theta},\boldsymbol{\phi}|\boldsymbol{d}) \propto P(\boldsymbol{d}|\boldsymbol{\theta},\boldsymbol{\phi})P(\boldsymbol{\theta},\boldsymbol{\phi})$. An independent prior distribution assumption is often used to facilitate the calculation for the posterior distribution (Higdon et al. 2004; Kennedy and O'Hagan 2001a): $P(\boldsymbol{\theta},\boldsymbol{\phi}) = P(\boldsymbol{\theta})P(\boldsymbol{\phi})$. The posterior distribution then becomes $Eq.(24)$.

$$P(\boldsymbol{\theta},\boldsymbol{\phi}|\boldsymbol{d}) \propto P(\boldsymbol{d}|\boldsymbol{\theta},\boldsymbol{\phi})P(\boldsymbol{\theta},\boldsymbol{\phi}) = P(\boldsymbol{d}|\boldsymbol{\theta},\boldsymbol{\phi})P(\boldsymbol{\theta})P(\boldsymbol{\phi}) \qquad Eq.(24)$$

The posterior distribution of calibration parameters can be estimated with full Bayesian analysis (Bayarri et al. 2007; Higdon et al. 2004) but it requires multidimensional integration and may be computationally prohibitive. As a substitute to intractable direct integral computation, the Markov chain Monte Carlo (MCMC) sampling method is widely used to estimate a posterior, which generates a sequence of realizations of the model parameters (Diaconis 2009). The alternative computationally efficient approach uses only the point estimates for the hyperparameters via Maximum Likelihood Estimation (MLE), as shown in $Eq.(25)$, instead of relying on full posterior distributions (Bayarri et al.

2007). The Quasi-Newton method (J. E. Dennis and Moré 1977) is often employed to optimize MLE (Qian and Wu 2008; Raissi, Perdikaris, and Karniadakis 2017b, 2017a; Rasmussen and Nickisch 2010).

$$\boldsymbol{\theta}^*_{\boldsymbol{MLE}}, \boldsymbol{\phi}^*_{\boldsymbol{MLE}} = \underset{\boldsymbol{\theta},\boldsymbol{\phi}}{\operatorname{argmax}}\, logP(\boldsymbol{d}|\boldsymbol{\theta},\boldsymbol{\phi}) \qquad Eq.(25)$$

If the computer model $f(\cdot)$ is a known mechanistic function with an explicit functional form, we can directly use the equation given instead of replacing it with a GP model. In this case, we train the discrepancy function with the difference between the experimental observations and the computer model output. Then the posterior distribution (Joseph and Yan 2015) is shown in $Eq.(26)$.

$$P(\boldsymbol{\theta},\boldsymbol{\phi}|\boldsymbol{d}) \propto \frac{1}{|\Sigma_\delta + \Sigma_\varepsilon|^{\frac{1}{2}}} \exp\left[(-\frac{1}{2}(\boldsymbol{d} - f(\boldsymbol{\theta}))'(\Sigma_\delta + \Sigma_\varepsilon)^{-1}(\boldsymbol{d} - f(\boldsymbol{\theta}))\right] P(\boldsymbol{\theta},\boldsymbol{\phi}) \qquad Eq.(26)$$

Figure 7 describes the model calibration procedure. Refer to (Kennedy and O'Hagan 2006) for full mathematical detail in the model calibration procedure.

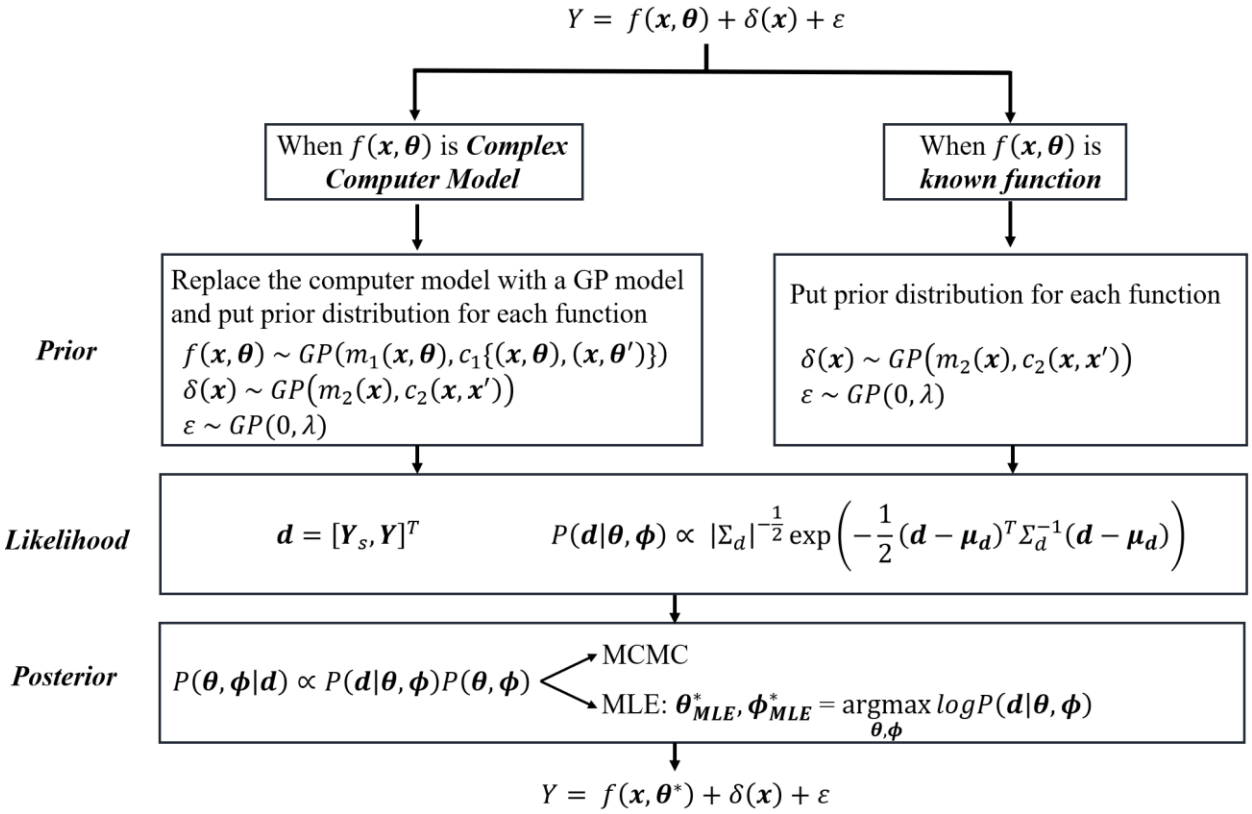

Figure 7. Model Calibration Procedure. Note that GP hyperparameter ϕ is omitted in $f(\boldsymbol{x},\boldsymbol{\theta})$ and $\delta(\boldsymbol{x})$ for simplicity.

One of the major advantages of model calibration is its ability to account for different sources of uncertainty (Arendt, Apley, and Chen 2012). In engineering and science applications, uncertainty is one

of the most important issues that every modeler needs to be careful to understand the dynamics correctly. Model calibration enables uncertainty quantification from different sources and helps make a proper decision by checking the degree of confidence and testing the reliability. The basic model calibration formulation $Eq.(20)$ incorporates different kinds of uncertainty from interpolation uncertainty through the $f(x,\theta)$ and $\delta(x)$ terms, parameter uncertainty through the estimation of the parameter set θ, to experimental uncertainty via ε. Another advantage in model calibration is its ability to incorporate domain knowledge in the form of prior settings for different parameters and discrepancy function. These informative prior settings can help generate an accurate posterior, and it is similar to the process of pre-training or transfer learning in NN modeling.

Informative Experimental Design is the other important concept that is enabled by model calibration (Chen et al. 2007). When a computer simulation or experiment is expensive, we need to decide which data to observe so as to produce the optimal design among many design candidates. In this case, the Bayesian approach in model calibration enables us to make decisions considering uncertainty in engineering design and find optimal design points under specific objectives. The Bayesian approach in model calibration is also advantageous in model validation when the computer simulation or the experiment is expensive (Chen et al. 2007; Lee, Kim, et al. 2019).

Challenges

While there are different advantages in the model calibration approach, there are some challenges worth noting. One of the main challenges in model calibration is that it is difficult to differentiate the effect of calibration parameters ($\boldsymbol{\theta}^*$) and the discrepancy function ($\delta(\boldsymbol{x})$). This is referred to as the Identifiability problem. Since we do not know which combination of calibration parameters and the discrepancy function is true, the system becomes non-identifiable (Arendt, Apley, and Chen 2012; Kennedy and O'Hagan 2001a).

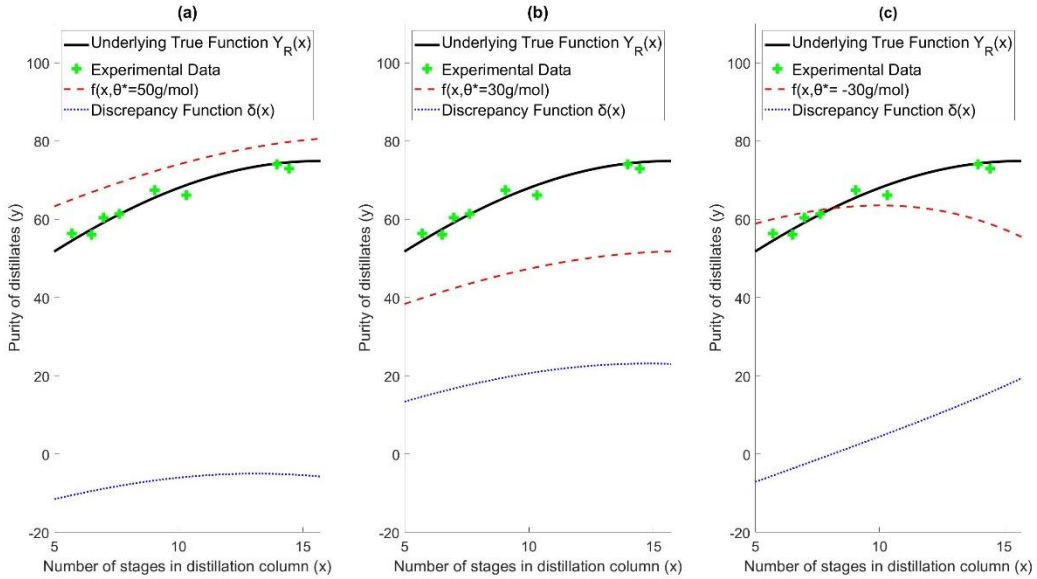

Figure 8. Identifiability issue in model calibration. (a), (b), and (c) predicts the same experimental data but there are different combinations of (posterior) calibration parameter $\boldsymbol{\theta}^*$ and the (posterior) discrepancy function $\delta(\boldsymbol{x})$

Figure 8 shows different combinations of calibration parameters and the discrepancy function that explain the same system. Calibrated model with discrepancy function ($f(\boldsymbol{x}, \boldsymbol{\theta}^*) + \delta(\boldsymbol{x}) + \varepsilon$) predicts the same output ($Y(\boldsymbol{x})$) well but the dynamics of $f(\boldsymbol{x}, \boldsymbol{\theta}^*)$ and $\delta(\boldsymbol{x})$ are different. It is shown (Arendt, Apley, and Chen 2012) that adding more simulation and experimental data for training the model does not solve the identifiability issue. To improve identifiability, an informative prior (accurate mean and low standard deviation) for the calibration parameter can be used (Bayarri et al. 2007; Liu, Bayarri, and Berger 2009). A specific functional form for the discrepancy function can also be employed for discrepancy function if we have sufficient prior knowledge (Joseph and Melkote 2009; Xiong et al. 2009). It is shown that identifiability is difficult but possible for some cases when proper prior for the calibration parameters and discrepancy function are set (Arendt, Apley, and Chen 2012). However, when the system is complex or not well understood, we may not have enough knowledge to set the informative prior for the calibration parameters and the discrepancy function.

Another important issue of model calibration is the lack of interpretability. In model calibration, the physical interpretation of calibration parameters is not recommended since the calibrated value can be far from the true value (Kennedy and O'Hagan 2001a). For example, if we set the molar weight of an unknown chemical as a calibration parameter ($\boldsymbol{\theta}$), the calibrated parameter ($\boldsymbol{\theta}^*$) can be negative, which is not physically valid in reality. Modelers may also be interested in determining important variables in the system or identifying potential causes for the discrepancy. While there exist different approaches (Lee,

747 Cho, and Lee 2019; Wipf and Nagarajan 2007; Yi, Shi, and Choi 2011; Piironen and Vehtari 2016;
748 Linkletter et al. 2006; Savitsky, Vannucci, and Sha 2011) for subset selection in GPR, the interpretation
749 of GP hyperparameter often produces a false interpretation of the system (Lin and Joseph 2019). The
750 incorporation of a discrepancy function and the model calibration parameter into GP modeling and the
751 traditional identifiability issues in model calibration makes it more difficult to interpret the model
752 calibration system correctly.

753 Prior settings on model parameters and the discrepancy function are critical for model performance
754 and interpretability in model calibration. While informative prior settings can help improve model
755 performance, inappropriate settings of a prior may lead to bad predictions and poor interpretability of the
756 system. However, specifying an appropriate prior is often challenging when we do not have enough prior
757 knowledge of the system. The effect of different priors of hyperparameters on the GPR prediction
758 performance is investigated in (Chen and Wang 2017) and concluded that the initial prior setting affects
759 the convergence of the posterior hyperparameters to the true value. Selecting appropriate model
760 discrepancy priors that capture missing physics in the system is also critical in model calibration. A
761 different form of model discrepancy prior is compared in (Ling, Mullins, and Mahadevan 2014) and
762 concluded that the calibrated parameters can be physically biased if we set inappropriate discrepancy
763 prior, and also affects the extrapolation capability. In addition, the choice of covariance function is a
764 critical problem since it profoundly affects the model performance (Rasmussen and Williams 2005;
765 Schulz, Speekenbrink, and Krause 2018). We need to choose the appropriate kernel that can capture the
766 system dynamics, but it is sometimes challenging if we do not have enough prior knowledge of the
767 system.

768 High computational cost is another important issue. In standard GPR, estimation of the posterior
769 includes inverting the matrix $[\Sigma + \sigma_n^2 \mathrm{I}]$, and the corresponding computational complexity is $O(N^3)$ where
770 N is the number of observations. Since the computational cost is cubically proportional to the number of
771 observations, the model becomes computationally expensive for a large dataset. In addition, the amount
772 of data required for reliable analysis grows exponentially as the dimensionality of data increases (Jin,
773 Chen, and Sudjitanto 2002; Higdon et al. 2004).

774 Although GPR is often considered a good smoothing interpolator, it has a limited capability in terms
775 of extrapolation. Different factors including kernel structure can affect the extrapolation ability of GPR
776 and model calibration. If the kernel is not comprehensive to capture the dynamics of data structure,
777 extrapolation performance would be poor (Wilson et al. 2013). For example, it has been shown that the
778 squared exponential kernel fails to capture the non-local structure of data (Bengio, Delalleau, and Roux
779 2005). The extrapolation capability also depends on the extendibility of the discrepancy function and the

780 confidence in the computer simulation (Higdon et al. 2004). How well the model inadequacy (between
781 experiment and computer model) is captured by the discrepancy function on the extrapolated region is an
782 important issue. Here, appropriate prior settings on the discrepancy function and the calibration parameter
783 play a significant role in determining extrapolation properties (Bayarri et al. 2007; Ling, Mullins, and
784 Mahadevan 2014). For example, it is possible to incorporate physics-based knowledge of the system on
785 the discrepancy function may also help improve extrapolation performance (Higdon et al. 2004), but this
786 is unlikely to be available for the majority of systems.

787 Different research is conducted to mitigate the drawbacks of model calibration. To increase
788 interpretability, statistical adjustment methods of engineering models are proposed with main effect
789 analysis (Joseph and Yan 2015) and based on empirical Bayes methods (Joseph and Melkote 2009). Also,
790 different frameworks for incorporating the discrepancy function have been introduced to deal with the
791 issue caused by the traditional model calibration, which simply treats the discrepancy function as an
792 additive term. For example, Plumlee (Plumlee 2017) proposes a discrepancy function prior that is
793 orthogonal to the gradient of the computer model, which shows improved behavior of posterior
794 distribution that matches with reality. Sargsyan et al. (Sargsyan, Najm, and Ghanem 2015) utilize model
795 parameterizations to capture the discrepancy instead of the additive discrepancy term. The development
796 of kernels that better reflects reality is an ongoing problem. (Duvenaud, Nickisch, and Rasmussen 2011)
797 introduce a Gaussian Process with additive kernels, which improves interpretability and the extrapolation
798 performance by considering a possible set of input interactions. Closed-form kernels that involve spectral
799 density with a Gaussian mixture are derived (Wilson and Adams 2013) for extrapolation and pattern
800 discovery. The physics-informed Kriging method (Yang, Tartakovsky, and Tartakovsky 2018) is
801 proposed, where different realizations of available data are used for constructing kernels without
802 assuming a specific form for the kernel function. Recently, active research is conducted to increase the
803 scalability of GP and process the large dataset by finding the pseudo-inputs for the dataset (Gramacy and
804 Apley 2015; L'Heureux et al. 2017; Yan and Qi 2010; McIntire, Ratner, and Ermon 2016; Snelson and
805 Ghahramani 2005; Liu et al. 2018). Further, the MapReduce framework (Dean and Ghemawat 2008) has
806 been investigated with the aim of accelerating the model calibration framework (Cai and Mahadevan
807 2017) when processing big datasets.

808 While there exist different advantages and limitations in model calibration, we believe that this
809 modeling technique can be effectively applied in process system engineering. Specifically, model
810 calibration techniques may be particularly useful in the field where the computer simulation or the
811 experiment is expensive, collected data is noisy, and the uncertainty plays a significant role in exploring
812 the system dynamics. We can observe how uncertainty is propagated during the process, which also helps
813 informative experimental design and model validation process when data acquirement is expensive.

4.5. Other techniques for merging mechanistic knowledge with data-driven models

In addition to the techniques discussed in this paper, there are other approaches that incorporate mechanistic information within data-driven models. A very common approach is feature engineering, which involves selecting or identifying (using Machine Learning), the appropriate features (or inputs) to be used to fit a Machine Learning model (Dong and Liu 2018). This approach is particularly influential in fields such as materials science, where large data sets are available without a-priori knowledge of the true physical descriptors that should be used to predict material properties (Butler et al. 2018). In PSE, it is typically likely that the important, controllable inputs to the models are already known. Associated with feature engineering, are techniques that perform subset selection for generalized linear regression, that aim to build generalizable parametric models with additive features that best describe the data (Cozad, Sahinidis, and Miller 2014; Wilson and Sahinidis 2017). These techniques lead to more interpretable models, when compared to nonparametric techniques such as NNs and GPs. However, these techniques are more scalable and effective if some a-priori knowledge is available to inform the initial "superset" of potential basis functions.

Another relevant research area to hybrid modeling is multifidelity modeling. Multifidelity techniques are used when multiple types of data and/or models are available for the same system (ranging from highly accurate to low-fidelity) and these and the correlation between them are jointly used to generate overall more accurate models (Peherstorfer, Willcox, and Gunzburger 2018). This approach is very relevant to hybrid submodeling and model calibration. The concept of developing models that are trained with various forms of fidelity has been often used for surrogate-based optimization, showing that learning from different sources of levels of fidelity of data can expedite the search for an optimal location. Multifidelity techniques have also been directly incorporated within Gaussian Process regression, or Kriging, via the co-Kriging algorithm that trains a GP model using various fidelity sources (Stein and Corsten 1991; Perdikaris, Venturi, and Karniadakis 2016; Perdikaris et al. 2015; Meng and Karniadakis 2020; Lee, Dietrich, et al. 2019).

Finally, there is number of noteworthy contributions that attempt to fuse the physics-based knowledge into data-driven modeling but could not be categorized within the methods outlined in this paper. Specifically, active research is conducted to incorporate physics-based knowledge within Gaussian Process Regression models. For the case where physics-based knowledge is in the form of linear differential equations, it has been shown that the GP can be constructed in a way to adhere such physics laws. The first approach includes the use of a specific covariance function that meets physical constraints. (Wahlström et al. 2013) embeds divergence and curl-free properties of the magnetic field by introducing divergence-free kernel. The second approach utilizes the well-known properties of the Gaussian process:

the linear transformation of a Gaussian process is also Gaussian, to embed the physics-based knowledge into GPR. (Raissi, Perdikaris, and Karniadakis 2017b, 2017a) utilize linear transformation of GP to infer parameters and learn solutions of the linear differential equations. Sarkka (Särkkä 2011) embeds physics-based knowledge as a prior to GPR as stochastic partial differential equations. Jidling et al. (Jidling et al. 2017) incorporate the linear operator constraints into the covariance function by introducing another linear operator that fulfills the constraint. Hegermann (Lange-Hegermann 2020) incorporates the boundary conditions on a linearly constrained GP. The third approach incorporates domain knowledge by generating different realizations of the physics-based model and construct mean and covariance function based on the collection of realizations without assuming any covariance structure (Yang, Tartakovsky, and Tartakovsky 2018; Tipireddy and Tartakovsky 2018).

5. Applications

5.1. Applications of Hybrid Sub-modeling

HSMs have been a been applied to a large and growing number of applications within chemical engineering, including modeling, monitoring, optimization, scale-up and control (von Stosch, Oliveira, et al. 2014). Applications also span most industries that fall under the umbrella of (bio)chemical engineering, including petrochemicals, metallurgy, wastewater treatment, papermaking and pharmaceuticals. Applications in these areas have led to a number of general proposed frameworks (Oliveira 2004; von Stosch et al. 2012; Abonyi, Madar, and Szeifert 2002; Sun et al. 2020) for HSMs, again many based on the original work of (Psichogios and Ungar 1992). For a summary of HSM implementations across different industrial sectors and applications, the reader should consult pertinent reviews (Zendehboudi, Rezaei, and Lohi 2018; von Stosch, Oliveira, et al. 2014; McBride, Sanchez Medina, and Sundmacher 2020; von Stosch, Davy, et al. 2014) as well as a recently published book (Glassey and Von Stosch 2018).

5.2. Applications of Physics-Informed Machine Learning

PI-ML models have been formulated to solve the forward and inverse problems in physics. In the inverse problem, the solution is given and the model estimates parameter values or conditions. In the forward problem, the model is given all parameters, initial conditions, and boundary conditions and it outputs the solution to the differential equation. Raissi et al. demonstrate their method with case studies such as the Burgers equation, Schrödinger equation and Navier-Stokes equation for fluid flow. The authors extend the PINN in in order to solve fractional advection-diffusion equations(Pang, Lu, and Karniadakis 2019). Here finite elements must be combined with the PINN approach as automatic differentiation via the chain rule since some terms results in an infinite series. In (Pakravan et al. 2020), a feed forward neural network finds PDE parameters which are fed to a custom layer that functions as a

PDE solver based off finite element methods. This layer is added to the overall NN, allowing for the model to be trained with the PDE solver embedded. In (Yang and Perdikaris 2018), deep probabilistic models are used to quantify uncertainty in the output of a PINN. This method can be used for verification analysis.

In robotics, physics-informed ML models have been applied to system control. In (Lutter, Ritter, and Peters 2019), neural networks are used to estimate inertial and force matrices of mechanical systems and the outputs of the neural network are combined into the physical equations of conservation (Lagrangian Mechanics). The output is automatically differentiated with a PID controller to give a control response. In power flow modeling, (Fioretto, Mak, and Van Hentenryck 2020) use the Lagrangian dual formulation to enforce Kirchoff's Current Law in NN models that act as surrogates for the AC optimal power flow problem. (Misyris, Venzke, and Chatzivasileiadis 2019) use the PINN formulation to simulate simple power system dynamics. The same methodology has applied to problems in geophysics (Qi Zhi He 2020; Kadeethum 2020) and cardiovascular modeling (Kissas 2020).

5.3. Applications of Model Calibration

The model calibration framework and the Bayesian inference is a very flexible and powerful tool, and it is widely used in various fields including energy simulation (Fabrizio and Monetti 2015; Kim and Park 2016; Manfren, Aste, and Moshksar 2013), optical lithography-based manufacturing process (Matsunawa, Yu, and Pan 2015), composite fuselage simulation (Wang et al. 2019), methane air chemistry (Sargsyan, Najm, and Ghanem 2015), CO_2 capture (Bhat et al. 2017; Kalyanaraman et al. 2015; Kalyanaraman et al. 2016; Li et al. 2017), fluid dynamics (Tagade, Jeong, and Choi 2013), and Li-ion cell operation (Tagade et al. 2016).

6. Computational Algorithms and Software Implementations

Due to the large range of potential HSM structures that are dependent on the scenario and the characteristics of the mechanistic SM, automated off-the shelf algorithms for HSM are not available. However, there are many recent developments in computational packages that significantly expedite the training of HSMs. Specifically, to fit a surrogate model (NN or GP) within a differential equation, any DE solver that tracks parameter sensitivities (i.e., gradients) can be used. However, parameter estimation of DEs using the sensitivity equations tends to become inefficient when the number of parameters is large (>100) (Rackauckas et al. 2018). Thus, recent software that calculates parameter sensitivities using automatic differentiation and adjoint-tracking are promising. Software with these capabilities are limited with notable implementation in Python, TensorFlow and PyTorch, and Julia via SciML. For more complex differential equation systems (i.e. PDEs) finite element software with ML extensions are gaining

momentum (Berg and Nystrom 2017; Mitusch, Funke, and Kuchta 2021). As the computational gains of these AD-based schemes are demonstrated, implementations in other scientific programming languages are anticipated to become available.

Several of the most notable algorithmic implementations are shown in Table 1 in the area of PI-ML. PINNs and PIGPs are typically trained with software that has native ML models and supports automatic differentiation, such as Tensorflow and Pytorch (Paszke et al. 2019; Abadi et al. 2016). Recently, packages have also been developed in Python (Haghighat and Juanes 2020a) and Julia (Rackauckas, Singhvi, et al. 2020) to create PINN models. In training, gradient-based optimization algorithms are used such as Newton's Method, stochastic gradient decent, Adam, and L-BFGS. Notably, the same software packages enabling PI-ML implementations are envisioned to automate the building HSMs.

For Model Calibration, multiple software packages are available and those are listed in Table 2. These can be used to generate posterior distributions for model parameters based on experiment and simulation data and make predictions.

Table 1. Algorithmic implementation of PI-ML algorithms

Name	Algorithm	Language	Reference
PINNs	Physics-Informed Neural Networks	Python	(Raissi 2019)
SciML	Physics-Informed NN for PDEs	Julia	(Rackauckas, Singhvi, et al. 2020)
Lagrangian Dual	Training PI-NNs using Lagrangian Duality	Python	(Freund 2004)
SciANN	Physics-Informed NNs for PDEs in Python	Python	(Haghighat and Juanes 2020a)
torchdiffeq	Neural Ordinary Differential Equations	Python	(Chen, Rubanova, et al. 2018)

Table 2. Model Calibration Packages

Name	Algorithm	Language	Reference
BACCO	Bayesian Calibration of Computer Codes	R	(Hankin 2005)
SAVE	Statistical Analysis of Computer Models	R	(Palomo, Paulo, and Garcia-Donato 2015)
PyMC3	Probabilistic Programming Library	Python	(Salvatier, Wiecki, and Fonnesbeck 2016)
GPM/SA	Gaussian Process Models for Simulation Analysis	Matlab	(Gattiker et al. 2015)
CaliCo	Bayesian Calibration	R	(Carmassi et al. 2018)
RobustCalibration	Full Bayesian Analysis for Model Calibration	R	(GU 2018)

It is worth observing here that many of the software implementations listed in the previous two tables are rely on open-source libraries that are supplemented with full-implementations of reproducible code. The past decade has seen the chemical engineering profession increasingly embrace the use of open-source code as a means of making software implementations more widely available and reproducible. It is likely that continued collaboration between engineers and colleagues in the computer science and machine learning communities will help perpetuate this mutually beneficial trend, accelerating the pace at which flexible, robust code becomes available to those who could benefit from its use. Echoing remarks in (Schäfer et al. 2020) a key goal of these collaborations should be to streamline the HM workflow such that not only the model training but also the data processing, model simulation and optimization can be performed in a single software environment. Such simplifications will naturally encourage more widespread industrial adoption of HMs.

7. Motivating Example

7.1. Case Study Description

In order to further elucidate how different formulations of hybrid models can be applied to systems familiar to the PSE audience, we will use a classic example in chemical engineering: an adiabatic plug flow reactor (PFR) (Snyder and Subramaniam 1994; Fogler 1999). In addition to being a fundamental unit operation in numerous chemical processes, the reactor system is associated with several modeling tasks that may be approached through a hybrid modeling lens. In the context of model calibration, kinetic and thermodynamic parameters inherent to the first-principles model must frequently be calibrated to reflect real data gathered from the reactor concentrations and temperatures. Similarly, data-driven models can be used to approximate certain parts of the model that may be not well described from physics, such as an equilibrium constant. Finally, the underlying differential equation model provides a basis for a PINN formulation, resulting in a full surrogate model that better generalizes to new points.

Specifically, we will look at the production of styrene from ethylbenzene (EB) with two side reactions (Rxns. 1-3).

$$Ethylbenzene \leftrightarrow Styrene + H_2 \qquad \text{(Rxn. 1)}$$

$$Ethylbenzene \rightarrow Benzene + Ethylene \qquad \text{(Rxn. 2)}$$

$$Ethylbenzene + H_2 \rightarrow Toluene + Methane \qquad \text{(Rxn. 3)}$$

The reversible rate law for Rxn 1, as well as the irreversible side reaction rate laws Rxn 2-3 are given below with corresponding units (kmol/ m^3/s):

$$r_1 = \rho(1-\phi)\exp\left(-0.08539 - \frac{10{,}925}{T}\right)\left(P_{EB} - \frac{P_{St}P_{H_2}}{K_{p1}}\right) \quad \text{(Eq. 27)}$$

$$r_2 = \rho(1-\phi)\exp\left(13.2392 - \frac{25{,}000}{T}\right)\left(P_{EB}\right) \quad \text{(Eq. 28)}$$

$$r_3 = \rho(1-\phi)\exp\left(0.2961 - \frac{11{,}000}{T}\right)\left(P_{EB}P_{H_2}\right) \quad \text{(Eq. 28)}$$

These rate laws can then be reformulated to mole balances on all of the species in the system, and combined with an energy balance to form the system of coupled differential equations shown below:

$$\frac{dF_{EB}}{dV} = -r_1 - r_2 - r_3 \quad \text{(Eq. 29)}$$

$$\frac{dF_{St}}{dV} = r_1 \quad \text{(Eq. 30)}$$

$$\frac{dF_{H_2}}{dV} = r_1 - r_3 \quad \text{(Eq. 31)}$$

$$\frac{dF_{Be}}{dV} = \frac{dF_{Et}}{dV} = r_2 \quad \text{(Eq. 32)}$$

$$\frac{dF_{Tol}}{dV} = \frac{dF_{Meth}}{dV} = r_3 \quad \text{(Eq. 33)}$$

$$\frac{dT}{dV} = \frac{-r_1\Delta H_{rxn1} - r_2\Delta H_{rxn2} - r_3\Delta H_{rxn3}}{F_{EB}Cp_{EB} + F_{St}Cp_{St} + F_{H_2}Cp_{H_2} + F_{Be}Cp_{Be} + F_{Et}Cp_{Et} + F_{Tol}Cp_{Tol} + F_{Meth}Cp_{Meth} + F_{Steam}Cp_{Steam}} \quad \text{(Eq. 34)}$$

In equations (29-34), F_i and P_i represent the flow rate and partial pressure of each chemical species *i*, respectively. T is the temperature and r the reaction rates. The independent variable V represents an increment of reactor volume and increases linearly with axial distance down the reactor unit. For an explanation of all other parameters in the model and their values, the reader is referred to the Appendix. Data for each framework is collected by simulating the mechanistic equations above for 6 separate runs, which vary using initial temperatures in the range $T \in [850,950]$ K and flow rates of the reacting species in the range $F \in [0,5]$ mol/m^3. In addition, an inert mixture of steam enters the reactor at a flow rate of 48 mol/m^3, acting as a heat sink. The reactor is assumed to operate at a total pressure P_T = 2.4 atm with negligible pressure drop. Thus partial pressures can be calculated as $P_i = (F_i/F_T)P_T$ where F_T is the sum of flow rates of all chemical species.

We will use this PFR system to demonstrate three methods: 1) Semi-parametric hybrid modeling, 2) Physics-Informed Learning Machine, 3) Model Calibration below.

7.2. Hybrid Modeling

To illustrate the difference in HSM approaches, we visit the ethylbenzene reactor example when insufficient mechanistic knowledge limits model performance. The data-driven model used for illustrating each of the HSM approaches is a feed-forward neural network with one hidden layer and 10 hidden nodes. However, other nonlinear models could be used. To explore the nuances between mechanism correction and estimation, we consider scenarios wherein missing thermodynamic or kinetic information prohibit a fully mechanistic modeling approach.

Unknown Energy Balance

First, we consider the scenario where a model for ethylbenzene conversion is available but fails to accurately capture system performance.

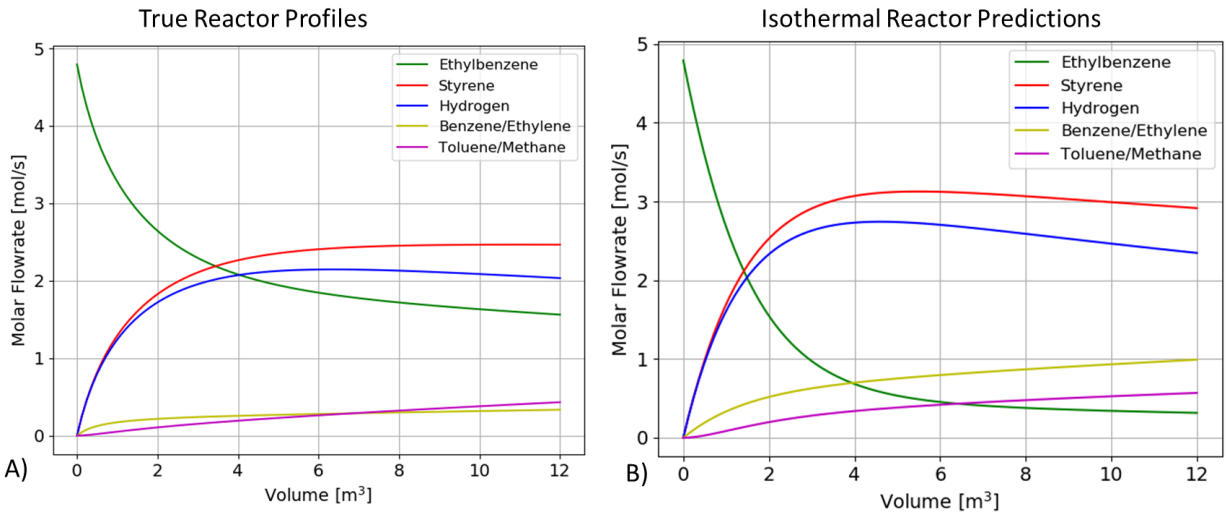

Figure 9. True (Left) vs low-fidelity (Right) simulation of ethylbenzene conversion to styrene.

Figure 9A shows the true progression of EB conversion in a tubular reactor assumed to be operating in plug-flow. The true temperature and flow rate profiles can be generated using the full set of equations found in the Case Study Description. In contrast, a low-fidelity mechanistic model which assumes isothermal conditions predicts the conversion profile in Figure 9B. The low-fidelity predictions are simulated using the same mechanistic equations as in the true model, except Equation 34 is replaced with $dT/dt = 0$. The task of this illustrative example is to improve the predictions of the low-fidelity mechanistic model.

A mechanism-estimation approach to model the unknown thermodynamics is to model the energy balance with a data-driven SM. This data-driven SM can be formulated as follows:

$$\frac{dT}{dt} = DD(T, F_k, \phi) \qquad \text{Eq. (36)}$$

Here the data-driven SM $DD()$ represents a neural network (NN) with 6 inputs, which include temperature T and the K=5 flow rates F_k of different chemical species in the reactor, where $k = 1, \ldots, K$. The NN has a single output dT/dt. The NN parameters are fitted by repeatedly integrating Eq. 36 with other mechanistic equations of the model, calculating the error in temperature predictions and updating the parameters of the NN using the error gradients until convergence. Note that this approach is feasible only by knowing the energy balance is the improperly formulated relationship.

An HSM for mechanism correction offers another approach to correcting the mechanistic model. The data-driven SM is trained and simulated separately from the mechanistic model. This framework follows Equations (2-3) covered in Section 4.2, which are repeated here for clarity.

$$\frac{dx_k}{dt} = f(x_k, c(t), \Theta) \qquad \text{Eq. (37)}$$

$$x_{\Delta t+t,k} = DD(x_{t,k}, c_t, \phi) \qquad \text{Eq. (38)}$$

Here $x_{t,k}$ represents each state variable k and time t, ϕ the parameters of the data-driven SM (i.e. the weights of a neural network) and Θ mechanistic parameters such as activation energy. $c(t)$ could potentially represent forcing variable such as a heating or cooling term, but these were not considering in this simple example. Training of the data-driven model is done using the flow rate and temperature predictions from the mechanistic model at position t as the inputs and the true concentrations/temperatures at position t + dt as the outputs. In total, the data-driven SM therefore has 6 inputs and 6 outputs.

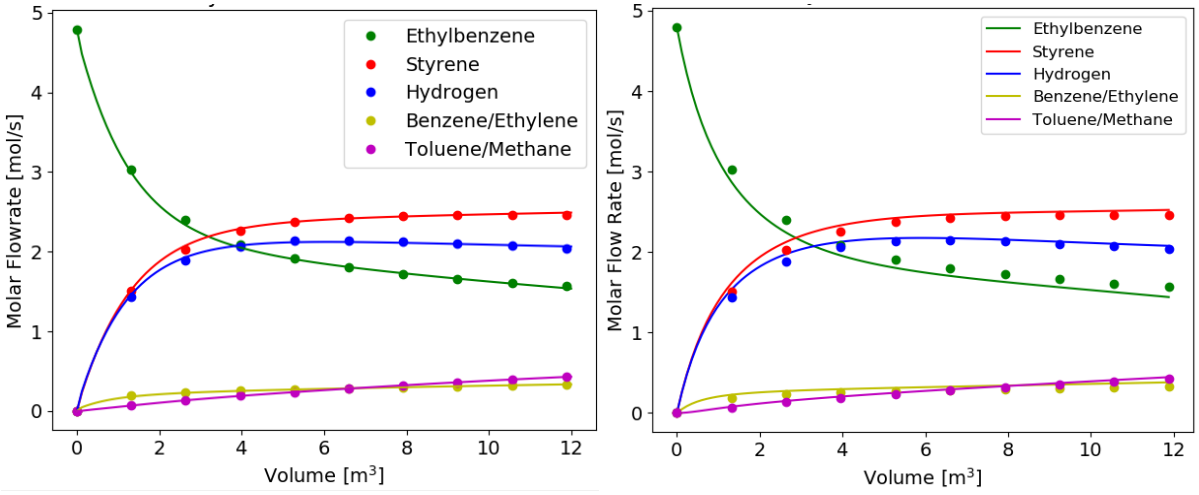

Figure 10. Regression of hybrid correction model (left) and hybrid estimation model (right). HM predictions represented by solid lines. Data used for regression represented by points.

The performance of the two methods is depicted in Figure 10. The predictions of both modeling methods are nearly identical and clearly improve the accuracy of the incomplete mechanistic model. The final mean squared error (MSE) between model predictions and data were much higher for the mechanism estimation approach than the mechanism hybrid correction approach—MSE = 3.396 mol/s vs MSE = 0.0775 mol/s, respectively. However, this should not be interpreted that the mechanism correction framework offers more accurate predictions in general. Rather, the estimation frameworks poorer predictions are a function of the more difficult optimization problem. The advantage of the estimation approach over the correction scheme is the clear function of the data-driven model. This may be useful if it is later desired to propose a mechanistic formula for correcting the low-fidelity model. The data-driven model predictions can be used to regress parameters of the mechanistic model separately from the other equations in the model.

Rate estimation

We also consider the scenario where the mechanistic model's kinetic rates (i.e., Equations 27-29) are unknown. Such situations may arise when a reaction mechanism is too complex to model via first principles knowledge or the kinetics have shifted due to catalyst deactivation or equipment degradation. Without a formulation for the kinetic rates, simulation of the remaining mechanistic equations yields no useful information, so a mechanism correction formulation is not viable. Nevertheless, using a mechanism estimation formulation, the rate relationships can be modeled with a data-driven model.

$$r = DD(T, F, \phi) \tag{39}$$

The model predicts the states via integration of the data-driven rate equations along with the known mechanistic equations. The data-driven model uses the 6 state variables as inputs and has three outputs, one for each rate. Data-driven parameters are trained by calculating the derivative of the loss function with respect to the parameters via automatic differentiation. It's worth noting that not all state variables contribute to the reaction rates, yet the neural network is still able to model the nonlinear rate law, ignoring the contribution of the insignificant state variables. However, feature selection techniques can be used to reduce the input dimensions and therefore the size of the data-driven model, which may also lead to mechanistic insight.

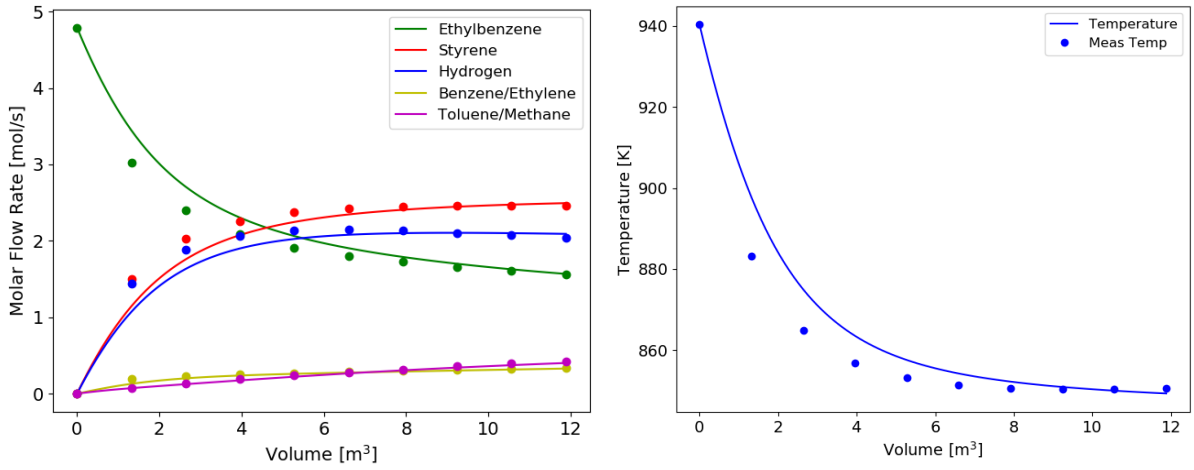

Figure 11. States (left) and temperature (right) predictions of the rates estimation HM. HM predictions represented by solid lines. Data used for regression represented by points.

The results of training are visualized in Figure 11. The final MSE of 2.438 mol/s between model predictions and data was comparable for cases of estimating the energy balance. For the examples of both mechanism correction and estimation, since all states are strongly dependent on temperature, the accuracy of the final model is limited to the range of temperature data used to train the data-driven model. However, the interpretability afforded by the kinetic terms in the hybrid estimation framework can serve to guide future experiments should a more generalizable model be required.

7.3. Physics-Informed Neural Network

In this Section, we illustrate the merits of using a PINN surrogate in lieu of a standard black-box NN model. Here we can use the underlying mechanistic knowledge of the system to improve ML predictions outside of training points for cases where interpolation or extrapolation is necessary. Using the PINN structure described in Section 4.3, we can train our model using sparse data as well as adherence to the underlying mechanistic equations given by the ODE system. Figure 12 below summarizes the inputs, outputs, and relevant physics relationships.

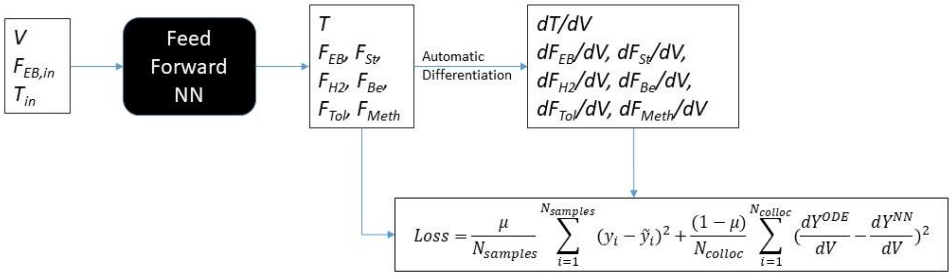

Figure 12. PINN Representation of Ethylbenzene Reactor with Loss Function Formulation

Importantly, the NN outputs are differentiated with respect to the input reactor coordinate (V). Using automatic differentiation, the resulting derivative values can be compared to Eq. 29-34. Critically, this can be done at unlabeled data points, allowing for good surrogates with very few labeled data points. This is shown in the loss function below, where labeled and unlabeled data are denoted N_{sample} and N_{colloc} respectively. Model predictions are given by y_i and ground truth by $\tilde{y}_i$.

$$Loss = \frac{\mu}{N_{samples}} \sum_{i=1}^{N_{samples}} (y_i - \tilde{y}_i)^2 + \frac{(1-\mu)}{N_{colloc}} \sum_{i=1}^{N_{colloc}} \left(\frac{dy_i}{dV} - \frac{d\tilde{y}_i}{dV}\right)^2 \quad \text{(Eq 40)}$$

For the sake of comparison, a classic NN model is fit to the same data under identical training conditions. Some results are given below to illustrate the effect of the physics-based loss term. First, the two models are tested under very sparse data conditions, where we only have training points for inlet, outlet and midway concentrations ($y(V = 0,6,12)$). For the PINN, we also select 100 collocation points randomly throughout the V domain. In Figure 13, the results for ethylbenzene concentration profiles for each model are depicted.

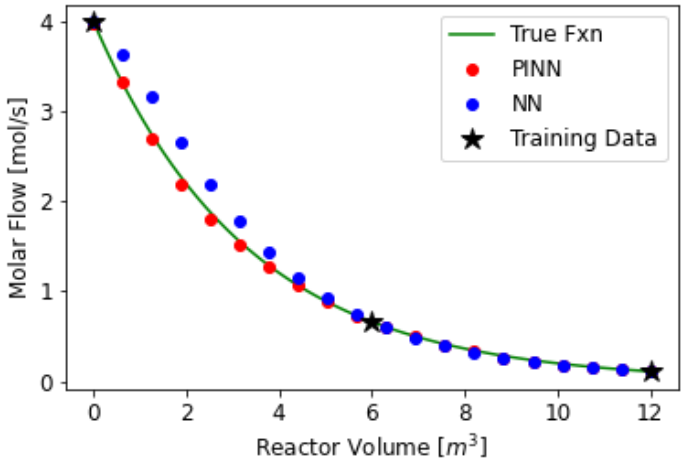

Figure 13. Comparison of true simulation prediction (data) with black-box NN (NN), and Physics-Informed NN (PINN) predictions for Interpolation between Training Points

While both models are able to fit the training data exactly, the PI-NN model has improved interpolating behavior than the black-box NN, especially in the more non-linear region ($V = [0,6]$). This is due to the physics constraint applied at collocation points throughout the domain. We can also design an experiment for extrapolation. Here we provide training data at $V = 2,6,10$ and predict over $V = [0,12]$.

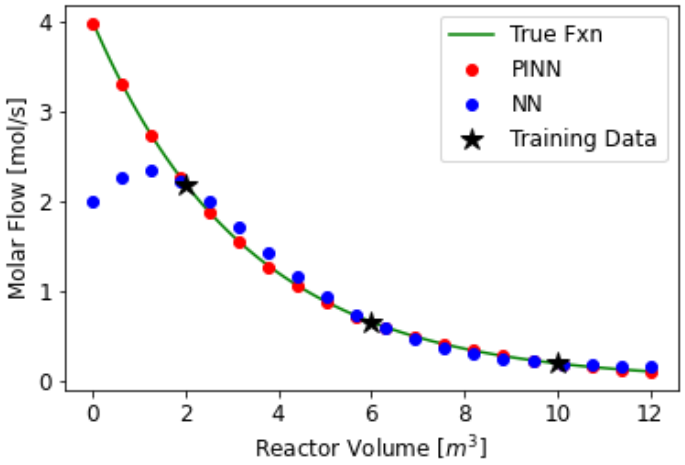

Figure 14. Comparison of true simulation prediction (data) with black-box NN (NN), and Physics-Informed NN (PINN) predictions for Extrapolation from Training Points

Again, both models are able to predict points where training data is present, but only the PI-NN can extrapolate outside that region because it has mechanistic knowledge applied at collocation points during training. From the above results, it is clear that the PINN approach offers advantages over the purely black-box approach, however, it requires that accurate physics-based information is available. In this example, we have assumed a fully-mechanistic model is available and have built a NN surrogate that is trained with embedded mechanistic knowledge. A similar approach can be applied even if the mechanistic knowledge is partially available. For example, the physics-constraint violation term could include violation from a mass-balance constraint, or an energy balance constraint, if those are the only mechanistic equations that are available. The difference between PINN and NN predictions would be further pronounced in applications with noisy data, as the differential constrain is essentially a regularization term that penalizes the objective function and reduces over-fitting and noise-fitting.

7.4. Model Calibration

Model calibration requires two datasets: ***Observed data*** from physical experiment or sensor (high-fidelity data) and ***Simulation data*** (low-fidelity data). Since it is difficult to obtain experimental data for the case study example, we first generate a dataset from the *full* Ethylbenzene reactor model and assume that it is the high-fidelity observed data. We generate the low-fidelity dataset by applying the simplifying assumption: $\frac{dT}{dV} = 0$ to the Ethylbenzene reactor model. We generate small data points (9 simulation data (Y_s) and 8 experimental observations (Y)) to show its performance on a limited dataset and introduce the Gaussian noise to consider the measurement error. We consider a single ***variable input*** (V: reactor volume) – single output ($F_{Ethylbenzene}$: molar flow rate of ethylbenzene) system with two calibration

parameters (ρ and b_1). Note that calibration parameters can be a physically interpretable parameter (ρ) or simply tuning parameter (b_1) which has little to no physical meaning. This process is depicted in Figure 15.

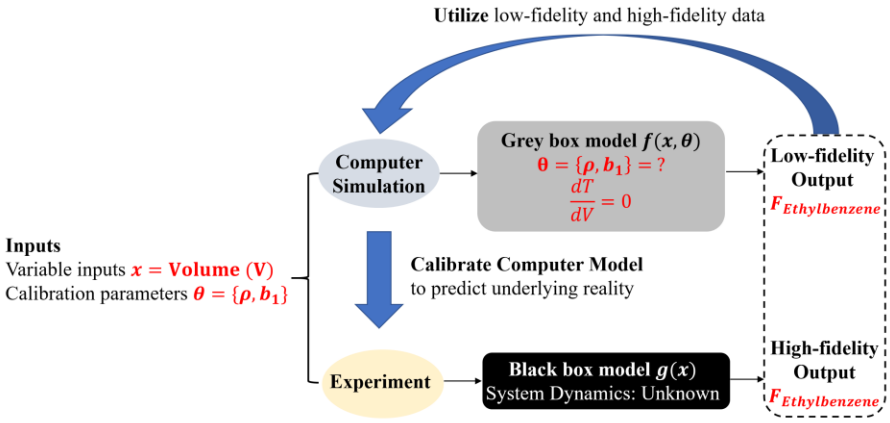

Figure 15. System Description for Model Calibration

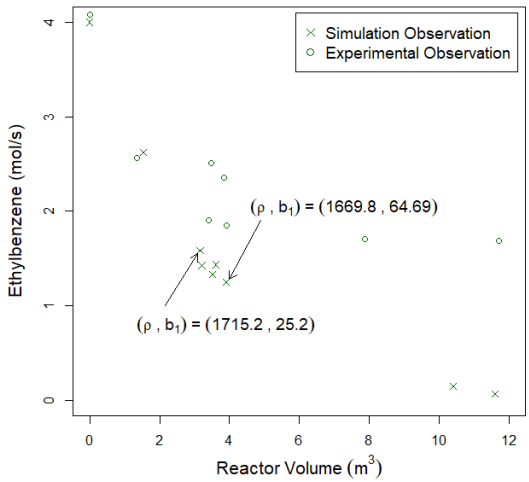

Figure 16. Different guesses of calibration parameters to run computer simulation in Ethylbenzene example

Figure 16 shows the different runs of computer simulation and the observed data. Note that **both** low-fidelity computer model output **and** the high-fidelity experimental data will be used for updating the computer model. We sample the data randomly within the input region so that we have a sparse/dense distribution within the range. It is important to note that we do not know the true value of calibration parameters $\theta^* = (\rho, b_1) = (2137, -17.34)$, but they are required to enter the values for these calibration parameters to run the computer simulation. Here, we assume that some prior knowledge is available regarding the calibration parameters. For example, experts may expect that there is a 95% chance of ρ be

in a range of [1700, 2100] and b_1 be in a range of [-50,50]. By utilizing this expert knowledge, we can sample (ρ, b_1) from the prior distribution $\rho \sim N(1900,\ 100^2)$ and $b_1 \sim N(0, 25^2)$ and run the computer simulation.

We follow the Modular approach (Arendt, Apley, and Chen 2012) when estimating calibration parameters and GP hyperparameters in model calibration for computational efficiency. The main point of this approach is that we estimate GP hyperparameters of the computer model and the discrepancy function in a sequential way by treating them separately. Firstly, we replace the computer model with GP and estimate GP hyperparameters of the computer model (ϕ_s^*) by considering the computer simulation data. Next, we fix the GP hyperparameters of the computer model, postulate a GP for the discrepancy function, and estimate the hyperparameters (ϕ_δ^*) of the discrepancy function. After obtaining the GP hyperparameters of the computer model and the discrepancy function, we use Bayesian inference to estimate the posterior distribution of calibration parameters (θ^*). We obtain the sample of posterior hyperparameters (ϕ^*) and calibration parameters (θ^*) via the MCMC algorithm. During the model calibration process, the variable input space and the calibration parameter space is scaled to the range of 0 to 1 for numerical stability (Hinton and Rasmussen 1997).

Figure 17 shows the model calibration result. The calibrated computer model predicts high-fidelity experimental observation with higher accuracy than the original computer simulation. The uncertainties of the model are also incorporated and can be seen as a 95% confidence interval. Note that uncertainty is larger in the region where we have sparse observations.

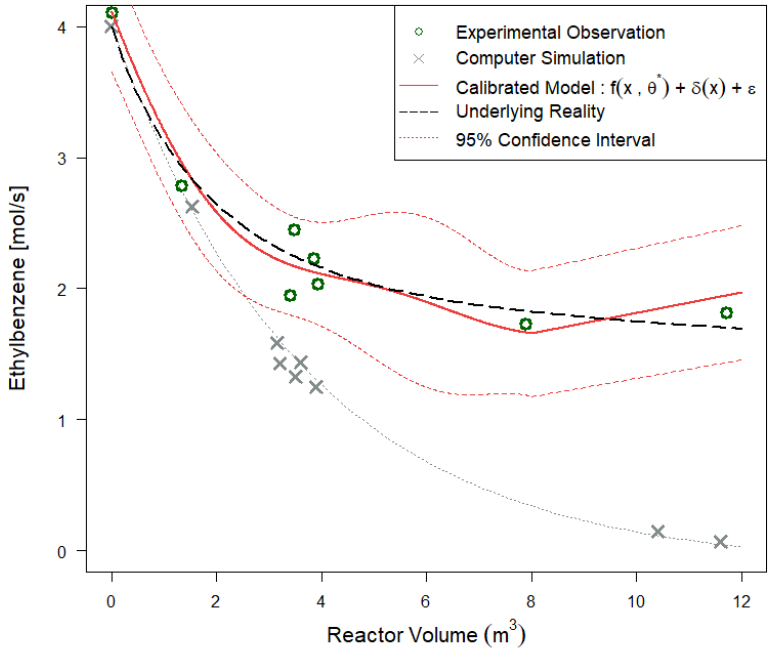

Figure 17. Model calibration result

In Figure 18, we observed how each term $f(\boldsymbol{x}, \boldsymbol{\theta})$ and $\delta(\boldsymbol{x})$ works separately. Here, $f(\boldsymbol{x}, \boldsymbol{\theta}^*)$ term (computer model with calibrated parameter) does not match with the original, complete computer simulation output because the value of calibration parameters does not converge to the true value. Moreover, even when we assume that the calibration parameters are converged to the true value, perfect agreement between the computer model and the real data ($Y = f(\boldsymbol{x}, \boldsymbol{\theta}^*)$) is not guaranteed because the computer model itself $f(\cdot)$ is the approximation to the reality with simplifying assumption ($\frac{dT}{dV} = 0$). It is possible that there remains a noticeable discrepancy between the computer model simulation and the experimental output if we do not consider the discrepancy $\delta(\boldsymbol{x})$ term. While model calibration can be done under the different model formulations (Drignei 2009; Lindgren, Tersing, and Domkin 2003; McFarland et al. 2008), Kennedy and O'Hagan (Kennedy and O'Hagan 2001a) model $Y(x) = f(\boldsymbol{x}, \boldsymbol{\theta}) + \delta(\boldsymbol{x}) + \varepsilon$ is the most popular framework for model calibration since it incorporates the discrepancy $\delta(\boldsymbol{x})$ in the model formulation.

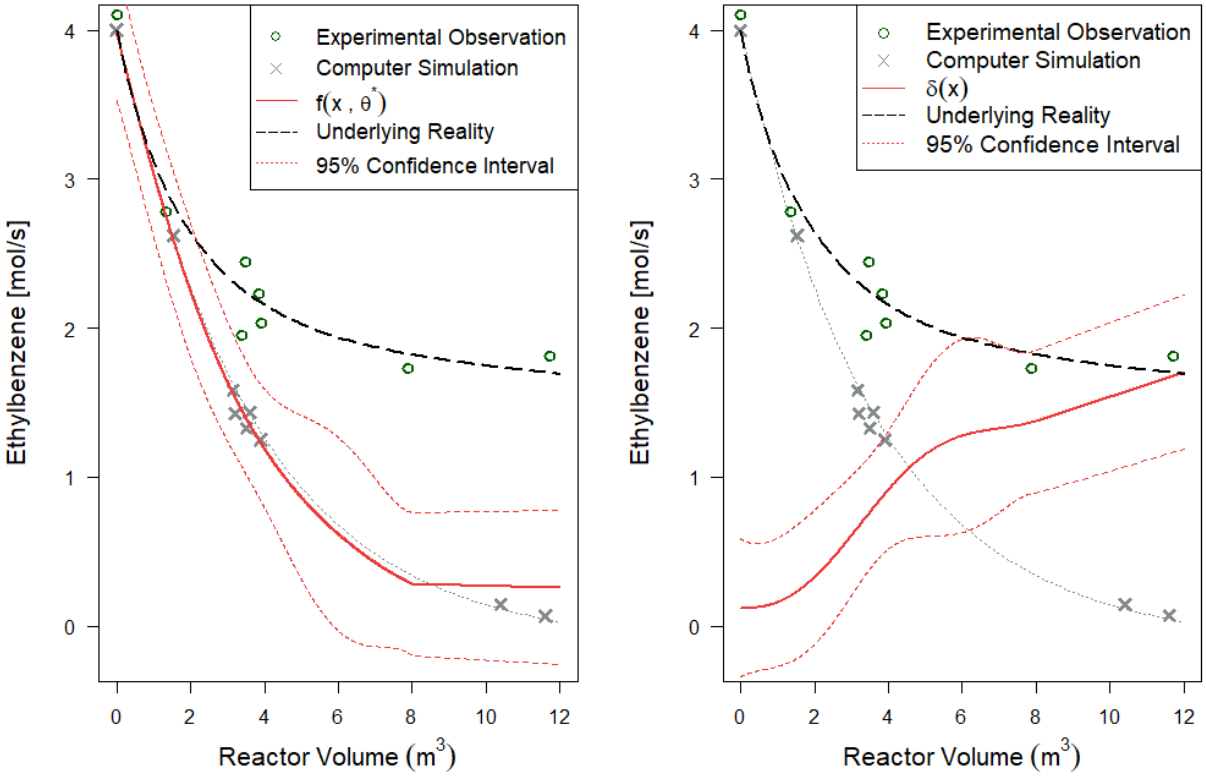

Figure 18. The behavior of each term in Model Calibration. The graphs show the contributions of the calibrated computer model $f(\boldsymbol{x}, \boldsymbol{\theta}^*)$ without discrepancy term $\delta(\boldsymbol{x})$ (left) along with the discrepancy function $\delta(\boldsymbol{x})$ only (right).

It is worth emphasizing that posterior mean of calibrated parameter values $(\rho_{calibrated}, b_{1,calibrated}) = (1846, 8.6)$ do not converge to the true value $(\rho_{true}, b_{1,true}) = (2137, -17.34)$. In fact, different combinations of calibration parameters $\theta^* = (\rho, b_1)$ and the discrepancy function $\delta(x)$ are possible for explaining the same system accurately. Therefore, it is dangerous to interpret the physical meaning out of it. For example, we cannot say that the catalyst bed density is $\rho_{calibrated} = 1846\ [kg/m^3\ pellet]$. We can see that this calibrated value is different from the true value of the catalyst bed density $\rho_{true} = 2137\ [kg/m^3\ pellet]$ in the real system.

8. Perspectives and Future Outlook

Comparison between HSM and model calibration

Having reviewed the current state of HSM and model calibration, this perspectives piece now offers us the unique opportunity to weigh the relative merits of the two. In certain respects, model calibration

can be summarized as the non-deterministic equivalent of hybrid models aimed at mechanism correction. They both seek to improve the performance of underperforming engineering model, generally by modeling the discrepancy between the engineering model and experimental data. Yet despite their parallel development and similar goals little attention has been given to juxtaposing the utility of the two options. Perhaps this can be attributed to their natural evolution within two distinct disciplines. Historically, HSM correction methods have primarily developed within the process systems field and have focused on correcting dynamic process models using a deterministic data-driven functions (e.g. neural networks, support vector machines, polynomials, etc.). In contrast, model calibration has developed primarily within statistics literature, focusing on correcting simulations both large and small with a non-deterministic Bayesian scheme based on Gaussian Processes.

The model calibration formulation differs from hybrid mechanism correction in that it incorporates the effect of measurement noise. Moreover, in model calibration, the parameters in the mechanistic model are updated (i.e. calibrated) at the same time the Gaussian Process for discrepancy function is fitted. For the case where the mechanistic model is too large for repeated simulation in an optimization routine, the mechanistic model can be replaced by a surrogate (usually a Gaussian process), which is trained with the GP representing the discrepancy. The parameters of the mechanistic model could be updated in a mechanism correction framework, but such cases are less frequent.

The obvious advantage of the Bayesian scheme is the straightforward interpretation of uncertainty information, streamlining conclusions that are based on variable sensitivity and prediction intervals. In contrast, the deterministic correction scheme has been favored in engineering circles largely for its lower-bar in terms of technical know-how—there being no requirement to specify prior distributions, sampling schemes, or interpreting posterior probabilities. In addition, by avoiding Gaussian Processes, the deterministic mechanism correction scheme avoids the cubically-increasing compute cost frequently cited to be a problem when calibrating models with large amounts of data. When modeling the discrepancy, both methods suffer from the same limitations on interpretability of the discrepancy model and low extrapolation accuracy. When these are demanded, the modeler may ultimately be required to revert to mechanistic approaches that modify the engineering model in ways that yield additional fundamental knowledge about the system (Joseph and Melkote 2009).

Extrapolation, Identifiability and Interpretability

For a model to be useful, it must predict with reasonable accuracy system conditions different from those considered during model training. These predictions may fall within the range of data used during training (i.e. interpolation) beyond the range of training data (i.e. extrapolation). Each of the hybrid frameworks have been demonstrated to offer superior predictions than a purely data-driven model, (Lee et

al. 2002; Van Can et al. 1996; Kennedy and O'Hagan 2001b) at least for interpolation tasks. In addition, in cases where the mechanistic model is incomplete or poorly formulated, simply refitting the mechanistic model has been shown to underperform a hybrid model based on model calibration (Kennedy and O'Hagan 2001b), correction (Aguiar and Filho 2001; Sun et al. 2020; Keskitalo and Leiviskä 2015), or estimation (Van Can et al. 1998; Georgieva, Meireles, and Feyo de Azevedo 2003). However, the improved generalizability of hybrid approaches over a purely data-driven approach should not be confused with the extrapolation potential of purely mechanistic models. When a mechanistic model is well-formulated, the physical meaning (i.e. interpretability) of its parameter values can be leveraged to hypothesize how the mechanistic model will perform on an entirely new set of conditions. For the majority of the hybrid frameworks in this survey, the final output model is a data-driven model. Thus, their generalizability is likewise confined to the conditions covered by the training data. For example, hybrid correction models are to known to have the same limited generalizability as a purely data-driven model (Van Can et al. 1996; van Can et al. 1997) and are thus rarely applied when good model extrapolation is required, such as in system optimization (Yang et al. 2020). Improving their generalizability requires either enlarging the range of conditions used to sample data for model-building or more efficient coverage of the conditions used for training—for example, through careful design of experiments.

A notable exception to the general view above is the hybrid estimation framework. The HM estimation scheme aims to limit the application of the data-driven model to only the data regions and model functions that require adjustment or mechanistic knowledge is missing. As a consequence, HSMs based on mechanism estimation have been shown to have some ability to predict accurately beyond observed data (i.e., extrapolation). This makes this formulation far more attractive to applications in optimization. However, this extrapolation potential is contingent upon HM predictions relying on the accurately specified mechanistic relationships when extrapolating (Braake, van Can, and Verbruggen 1998; Van Can et al. 1998). A useful illustration of this principle can be found in (Yang, Martin, and Morris 2011). These authors modeled an unknown kinetic rate in the toluene nitration process via a data-driven SM while assuming a mechanistic SM for mass transfer is known. Since the kinetic rate was not dependent on the concentration or volume ratio of toluene these process variables could be changed without a significant decrease in the HM accuracy. Nevertheless, there are cases where the extrapolation accuracy of the hybrid estimation model may resemble that of a purely data-driven approach. For example, if a process model consists of highly coupled nonlinear conservation balance equations, separating the effects of data-driven and mechanistic SMs is not always obvious or possible. For such scenarios, more work is needed to discover whether there is any advantage in estimating some

relationships with mechanistic knowledge or whether a purely data-driven dynamic model will perform equally well.

In the case of Physics-Informed ML, while embedded physical knowledge cannot eliminate these challenges, several literature studies ((Raissi, Perdikaris, and Karniadakis 2019),(Kim, Lee, et al. 2020; Fraces, Papaioannou, and Tchelepi 2020)) demonstrate appreciable improvements for the extrapolation ability of the PI-ML models by acting as a regularization term. In (Haghighat and Juanes 2020b), a simple PDE model example clearly illustrates this fundamental trade-off. Here we use a NN to fit an "unknown" function: $f(x,y) = \sin(x)\sin(y)$. We know the data should follow this underlying PDE: $f_{xx} + f_{yy} + 2f = 0$, therefore, we can formulate a NN that learns the function solely from data points and a PINN which embeds the PDE knowledge. We only have data for the domain x,y = [-π,π] but want to make predictions over [-2π,2π]. Using identical NN structures, data, and training epochs, the performance of both methods is compared in Figure 19.

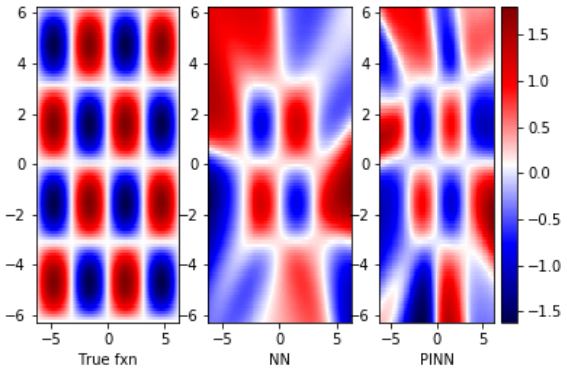

Figure 19. Extrapolation Ability of PINNs (Haghighat and Juanes 2020a)

In both, the functional representation is distorted the farther away from the training domain (shown as inner four squares above). However, the PINN clearly better represents the pattern outside of the domain compared to the NN. A less obvious point is that the NN does better within the training regime because it is overfitting. Another observation frequently absent in the literature is that not all modes of extrapolation are equivalent. While in the above example, PINNs show good generalization beyond the range of spatial coordinates used during training, in general their ability to rigorously extrapolate to new situations, say with different initial or boundary conditions, is limited. This observation can be explained by the use of independent variables (often spatial and time coordinates), which confines its validity to a particular spatio-temporal trajectory. Further research is needed to systematically explore which mathematical functions are most helpful for regularization/extrapolation and if non-differential functions can provide similar benefits.

Uncertainty Quantification

Developing standard methods for quantifying the uncertainty of the assumptions in hybrid frameworks remains an open problem. Studies that characterize the reliability, identifiability, and sensitivity of HSMs would be a useful step in this direction. Optimization of hybrid models using uncertainty information was tackled in (Kahrs and Marquardt 2007) for algebraic systems and recently extended to dynamic systems by (Bae et al. 2020). They established two criteria to measure the validity space: 1) a convex-hull criteria created a region around the measured data to define a space of trusted inputs to the hybrid model and 2) a confidence interval criteria constrained the HM's final predictions. Both could be added to the optimization formulation, either as constraints or as part of the objective function, to limit the predictions of the data-driven model in regions where the data-driven model was required to extrapolate. However, as noted by the authors in (Kahrs and Marquardt 2007), the convex-hull fails to be informative when data is sparse. A more nonlinear characterization of the data range used for training was demonstrated through a clustering technique (Simutis et al. 1995; Teixeira et al. 2005). By assuming a Gaussian distribution of cluster points, Simutis et al. were able define a criterion for the extrapolation of both the input and output space. More recently, (Pinto et al. 2019) used a bagging technique to fit an ensemble of NNs, which enabled construction of a confidence interval for the final HM. Another promising approach is to use topological data analysis along with a nonlinear classifier to construct validity regions. In (Schweidtmann et al. 2021), Schweidtmann's group showed how these validity regions could constrain data-driven models during optimization, suggesting that the method would be similar for hybrid models. Additional research is needed to understand how to apply these UQ techniques for model adaption and optimization for models with multiple SMs. Further progress in characterizing uncertainty for hybrid submodeling frameworks will likely result from employing statistical techniques previously developed from the Bayesian perspective of the model calibration framework.

In fact, the model calibration scheme may offer the most comprehensive treatment of uncertainty. This may be useful in, for example, an experimental design program, especially when the simulation or experiment is expensive. By using a Bayesian-based approach with design validation metrics (Chen et al. 2007), the modeler may determine what system conditions will yield the most informative design under a framework similar to model calibration. Recently, the model calibration with the emphasis on the uncertainty is discussed under three problems the forward, inverse, and the validation problem (Lee, Kim, et al. 2019). Another use case where the calibration emphasis on distribution can be advantageous is when we can take domain knowledge into account as a form of prior settings for different parameters and discrepancy function. This offers a more straightforward interpretation of the discrepancy model and hyperparameters based on original confidence in the mechanistic model parameters. However, as

previously stated, when little information is known *a priori*, setting prior distributions can be difficult. If little prior information is available, many modelers may choose a simple prior distribution (e.g., uniform distribution) (Chen and Wang 2017).

Hard Constraints vs Soft Constraints in the physic-embedded learning

One important distinction to draw from existing literature in PI-ML and for future work in this area is that of soft and hard constraints. Simply stated, a soft constraint is one that is penalized in the optimization algorithm but does not need to be satisfied in order to converge to an optimal solution. Hard constraints restrict the feasible space of the objective function and must be fully satisfied for the algorithm to converge. Strengths of the soft constraint include simple implementation, compatibility with ML training algorithms, robustness to noise in training data, and the ability to enforce physical knowledge that may not be fully accurate but still improves model performance. Soft constraints fit naturally within the ML field as a regularization technique for deep models. Their weakness in a traditional context is a lack of constraint guarantees and fuzziness about their full contribution to a model.

A PI-ML model with hard constraints could provide formal guarantees which may be attractive in many applications, but it is much less clear how to formulate these problems without losing the advantages of ML models. The Lagrangian-dual method referenced (Fioretto, Mak, and Van Hentenryck 2020; Fioretto et al. 2020) in Section 5.3 is one way to enforce hard constraints, if the Karush-Kuhn-Tucker (KKT) conditions can be satisfied. However, it is not guaranteed that a feasible solution will be found and the constraints are applied to discrete input points rather than a continuous domain. Using a fine input grid to approximate a continuous constraint results in a very large problem that may be overly constrained to find the optimum. While some studies (Mohan et al. 2020; Fioretto, Mak, and Van Hentenryck 2020; Zhang et al. 2018) have shown how this may work in practice, much work still needs to be done to clarify where hard constraints fit within this field.

9. Conclusions

This perspective piece has reviewed historical trends and recent contributions towards developing frameworks that merges physics-based and data-driven knowledge for modeling tasks, especially for processes generating dynamic data. From the above summary, we make the following observations:

- While not a panacea for all modeling tasks, hybrid methods may be key to reliable, fast process characterization when either first-principles alone or data alone cannot offer robust predictive power.

- In recent years software for implementing HMs has in many ways matured enough for user-friendly implementation of HMs.
- While HMs outperform purely data-driven approaches, they are more limited in their ability to reveal additional mechanistic insight about a system than a purely first-principles approach. Modelers should take care when extrapolating that the data-driven component does not overly influence the output of the model. Notwithstanding, HM approaches that yield mechanistic insights are certainly welcome.
- Separating mechanisms into different submodels was envisioned early, but no general framework for quantifying the contributions of each SM is available. Efficient, general frameworks for quantifying the validity (i.e., uncertainty) of models with multiple SMs would help define which and how many SMs to use for a given system.
- The Bayesian component of model calibration can be a useful tool when exploring the data space required for correcting mechanistic computer simulations. Probabilistic info can be leveraged to reduce the number of experiments and qualify the validity of the calibrated model. However, less work has been done on Bayesian analysis of dynamic systems. Studies that streamline this analysis would certainly encourage the calibration framework's regular use in the chemical processes field, where dynamic, and often unexpected, changes in the system are commonplace.

Declaration of Competing Interests

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Author Contributions Statement

WB, JK, ZK and FB wrote and edited manuscript. CL revised and edited the manuscript. FB supervised the writing and secured funding.

Funding

The authors gratefully acknowledge funding received from RAPID/NNMI Grant #GR10002225, Georgia Tech start-up grant and NSF CBET grants (1336386 and **1944678**).

Disclaimer. Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC., a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525. This paper describes objective technical results and analysis. Any subjective

views or opinions that might be expressed in the paper do not necessarily represent the views of the U.S. Department of Energy or the United States Government.

10. References

Abadi, Martín, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, and Matthieu Devin. 2016. 'Tensorflow: Large-scale machine learning on heterogeneous distributed systems', *arXiv preprint arXiv:1603.04467*.

Abonyi, Janos, Janos Madar, and Ferenc Szeifert. 2002. 'Combining First Principles Models and Neural Networks for Generic Model Control.' in Rajkumar Roy, Mario Köppen, Seppo Ovaska, Takeshi Furuhashi and Frank Hoffmann (eds.), *Soft Computing and Industry: Recent Applications* (Springer London: London).

Agarwal, Mukul. 1997. 'Combining neural and conventional paradigms for modelling,prediction and control', *International Journal of Systems Science*, 28: 65-81.

Aguiar, Helena Cristina, and Rubens Maciel Filho. 2001. 'Neural network and hybrid model: a discussion about different modeling techniques to predict pulping degree with industrial data', *Chemical Engineering Science*, 56: 565-70.

Arendt, Paul, Daniel Apley, and Wei Chen. 2012. 'Quantification of Model Uncertainty: Calibration, Model Discrepancy, and Identifiability', *Journal of Mechanical Design*, 134: 100908.

Arnold, Florian, and Rudibert King. 2021. 'State–space modeling for control based on physics-informed neural networks', *Engineering Applications of Artificial Intelligence*, 101: 104195.

Bae, Jaehan, Hye ji Lee, Dong Hwi Jeong, and Jong Min Lee. 2020. 'Construction of a Valid Domain for a Hybrid Model and Its Application to Dynamic Optimization with Controlled Exploration', *Industrial & Engineering Chemistry Research*, 59: 16380-95.

Bayarri, Maria J., James O. Berger, Rui Paulo, Jerry Sacks, John A. Cafeo, James Cavendish, Chin-Hsu Lin, and Jian Tu. 2007. 'A Framework for Validation of Computer Models', *Technometrics*, 49: 138-54.

Baydin, Atılım Günes, Barak A. Pearlmutter, Alexey Andreyevich Radul, and Jeffrey Mark Siskind. 2017. 'Automatic differentiation in machine learning: a survey', *J. Mach. Learn. Res.*, 18: 5595–637.

Bengio, Y., Olivier Delalleau, and Nicolas Roux. 2005. *The Curse of Highly Variable Functions for Local Kernel Machines*.

Berg, Jens, and K. Nystrom. 2017. 'Neural network augmented inverse problems for PDEs', *arXiv: Machine Learning*.

Bhat, K. Sham, David S. Mebane, Priyadarshi Mahapatra, and Curtis B. Storlie. 2017. 'Upscaling Uncertainty with Dynamic Discrepancy for a Multi-Scale Carbon Capture System', *Journal of the American Statistical Association*, 112: 1453-67.

Bhosekar, Atharv, and Marianthi Ierapetritou. 2018. 'Advances in surrogate based modeling, feasibility analysis, and optimization: A review', *Computers & Chemical Engineering*, 108: 250-67.

Bishop, Christopher M. 2006. *Pattern recognition and machine learning* (springer).

Bollas, G. M., S. Papadokonstadakis, J. Michalopoulos, G. Arampatzis, A. A. Lappas, I. A. Vasalos, and A. Lygeros. 2003. 'Using hybrid neural networks in scaling up an FCC model from a pilot plant to an industrial unit', *Chemical Engineering and Processing: Process Intensification*, 42: 697-713.

Braake, H. A. B. te, H. J. L. van Can, and H. B. Verbruggen. 1998. 'Semi-mechanistic modeling of chemical processes with neural networks', *Engineering Applications of Artificial Intelligence*, 11: 507-15.

Butler, Keith T, Daniel W Davies, Hugh Cartwright, Olexandr Isayev, and Aron Walsh. 2018. 'Machine learning for molecular and materials science', *Nature*, 559: 547-55.

Cai, Guowei, and Sankaran Mahadevan. 2017. 'Model Calibration with Big Data.' in.

Carmassi, Mathieu, Pierre Barbillon, Matthieu Chiodetti, Merlin Keller, and Eric Parent. 2018. *CaliCo: a R package for Bayesian calibration*.

Chen, Libei, Yves Hontoir, Dexian Huang, Jie Zhang, and A. Julian Morris. 2004. 'Combining first principles with black-box techniques for reaction systems', *Control Engineering Practice*, 12: 819-26.
Chen, Liqun, Shuyang Dai, Yunchen Pu, Erjin Zhou, Chunyuan Li, Qinliang Su, Changyou Chen, and Lawrence Carin. 2018. "Symmetric variational autoencoder and connections to adversarial learning." In *International Conference on Artificial Intelligence and Statistics*, 661-69. PMLR.
Chen, Ricky T. Q., Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. 2018. "Neural Ordinary Differential Equations." In *arXiv e-prints*.
Chen, Wei, Ying Xiong, Kwok-Leung Tsui, and Shuchun Wang. 2007. 'A Design-Driven Validation Approach Using Bayesian Prediction Models', *Journal of Mechanical Design*, 130.
Chen, Yingjie, and Marianthi Ierapetritou. 2020. 'A framework of hybrid model development with identification of plant-model mismatch', *AIChE Journal*, 66: e16996.
Chen, Zexun, and Bo Wang. 2017. 'How priors of initial hyperparameters affect Gaussian process regression models', *Neurocomputing*, 275.
Cozad, Alison, Nikolaos V Sahinidis, and David C Miller. 2014. 'Learning surrogate models for simulation-based optimization', *AIChE Journal*, 60: 2211-27.
Cybenko, George %J Mathematics of control, signals, and systems. 1989. 'Approximation by superpositions of a sigmoidal function', 2: 303-14.
de Azevedo, C. Rodrigues, J. Peres, and M. von Stosch. 2015. 'An efficient method for the numerical integration of measured variable dependent ordinary differential equations', *Engineering Applications of Artificial Intelligence*, 38: 24-33.
Dean, Jeffrey, and Sanjay Ghemawat. 2008. 'MapReduce: simplified data processing on large clusters', *Commun. ACM*, 51: 107–13.
Deb, Chirag, Fan Zhang, Junjing Yang, Siew Eang Lee, and Kwok Wei Shah. 2017. 'A review on time series forecasting techniques for building energy consumption', *Renewable and Sustainable Energy Reviews*, 74: 902-24.
Dhillon, Anamika, and Gyanendra K Verma. 2020. 'Convolutional neural network: a review of models, methodologies and applications to object detection', *Progress in Artificial Intelligence*, 9: 85-112.
Diaconis, Persi. 2009. 'The Markov Chain Monte Carlo Revolution', *Bulletin of the American Mathematical Society*, 46: 179textendash205.
Dissanayake, M. W. M. G., and N. Phan-Thien. 1994. 'Neural-network-based approximations for solving partial differential equations', *Communications in Numerical Methods in Engineering*, 10: 195-201.
Dong, Guozhu, and Huan Liu. 2018. *Feature engineering for machine learning and data analytics* (CRC Press).
Drignei, Dorin. 2009. 'A kriging approach to the analysis of climate model experiments', *Journal of Agricultural, Biological, and Environmental Statistics*, 14: 99.
Duarte, Belmiro, P. M. Saraiva, and Constantinos Pantelides. 2004. "Combined Mechanistic and Empirical Modelling." In *International Journal of Chemical Reactor Engineering*.
Duvenaud, David, Hannes Nickisch, and Carl Rasmussen. 2011. 'Additive Gaussian Processes'.
Fabrizio, Enrico, and Valentina Monetti. 2015. 'Methodologies and Advancements in the Calibration of Building Energy Models', *Energies*, 8: 2548-74.
Fiedler, Bernold, and Andreas Schuppert. 2008. 'Local identification of scalar hybrid models with tree structure', *IMA Journal of Applied Mathematics*, 73: 449-76.
Fioretto, Ferdinando, Terrence WK Mak, and Pascal Van Hentenryck. 2020. "Predicting AC optimal power flows: Combining deep learning and lagrangian dual methods." In *Proceedings of the AAAI Conference on Artificial Intelligence*, 630-37.

Fioretto, Ferdinando, Pascal Van Hentenryck, Terrence WK Mak, Cuong Tran, Federico Baldo, and Michele %J arXiv preprint arXiv:.09394 Lombardi. 2020. 'Lagrangian Duality for Constrained Deep Learning'.

Fogler, H. Scott. 1999. *Elements of chemical reaction engineering* (Third edition. Upper Saddle River, N.J. : Prentice Hall PTR, [1999] ©1999).

Fraces, Cedric G, Adrien Papaioannou, and Hamdi %J arXiv preprint arXiv:.05172 Tchelepi. 2020. 'Physics Informed Deep Learning for Transport in Porous Media. Buckley Leverett Problem'.

Freund, Robert M. 2004. 'Applied Lagrange duality for constrained optimization'.

Gattiker, James, Kary Myers, Brian Williams, Dave Higdon, Marcos Carzolio, and Andrew Hoegh. 2015. 'Gaussian Process-Based Sensitivity Analysis and Bayesian Model Calibration with GPMSA.' in.

Georgieva, P., M. J. Meireles, and S. Feyo de Azevedo. 2003. 'Knowledge-based hybrid modelling of a batch crystallisation when accounting for nucleation, growth and agglomeration phenomena', *Chemical Engineering Science*, 58: 3699-713.

Gibert, K., M. Sànchez-Marrè, and J. Izquierdo. 2016. 'A survey on pre-processing techniques: Relevant issues in the context of environmental data mining', *AI Communications*, 29: 627-63.

Glassey, Jarka, and Moritz Von Stosch. 2018. "Hybrid Modeling in Process Industries." In. Milton: CRC Press.

Goebel, R., R. G. Sanfelice, and A. R. Teel. 2009. 'Hybrid dynamical systems', *IEEE Control Systems Magazine*, 29: 28-93.

Goldberg, Yoav %J Synthesis lectures on human language technologies. 2017. 'Neural network methods for natural language processing', 10: 1-309.

Gorbach, N. S., A. A. Bian, B. Fischer, S. Bauer, and J. M. Buhmann. 2017. 'Model Selection for Gaussian Process Regression', *Pattern Recognition (Gcpr 2017)*, 10496: 306-18.

Gramacy, Robert B., and Daniel W. Apley. 2015. 'Local Gaussian Process Approximation for Large Computer Experiments', *Journal of Computational and Graphical Statistics*, 24: 561-78.

Grossberg, Stephen %J Neural networks. 1988. 'Nonlinear neural networks: Principles, mechanisms, and architectures', 1: 17-61.

GU, Mengyang. 2018. 'Robust calibration of Imperfect Mathematical Models', *R package version 0.5.0.*

Gusmão, Gabriel S, Adhika P Retnanto, Shashwati C da Cunha, and Andrew J Medford. 2020. 'Kinetics-Informed Neural Networks', *arXiv preprint arXiv:2011.14473*.

Haghighat, Ehsan, and Ruben Juanes. 2020a. 'SciANN: A Keras wrapper for scientific computations and physics-informed deep learning using artificial neural networks', *arXiv preprint arXiv:2005.08803*.

Haghighat, Ehsan, and Ruben %J arXiv preprint arXiv:.08803 Juanes. 2020b. 'SciANN: A Keras wrapper for scientific computations and physics-informed deep learning using artificial neural networks'.

Hajirahimi, Z., and M. Khashei. 2020. 'Sequence in Hybridization of Statistical and Intelligent Models in Time Series Forecasting', *Neural Processing Letters*: 1-21.

Hankin, Robin K. S. 2005. 'Introducing BACCO, an R Bundle for Bayesian Analysis of Computer Code Output', *Journal of Statistical Software*, 14: 1-21.

Higdon, Dave, Marc Kennedy, James C. Cavendish, John A. Cafeo, and Robert D. Ryne. 2004. 'Combining Field Data and Computer Simulations for Calibration and Prediction', *SIAM Journal on Scientific Computing*, 26: 448-66.

Hinton, Geoffrey E., and C. Rasmussen. 1997. "Evaluation of gaussian processes and other methods for non-linear regression." In.

Hochreiter, Sepp. 1998. 'The vanishing gradient problem during learning recurrent neural nets and problem solutions.', *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*.

J. E. Dennis, Jr., and Jorge J. Moré. 1977. 'Quasi-Newton Methods, Motivation and Theory', *SIAM Review*, 19: 46-89.
Jagtap, Ameya D., Kenji Kawaguchi, and George Em Karniadakis. 2020. 'Adaptive activation functions accelerate convergence in deep and physics-informed neural networks', *Journal of Computational Physics*, 404: 109136.
Jagtap, Ameya D., Ehsan Kharazmi, and George Em Karniadakis. 2020. 'Conservative physics-informed neural networks on discrete domains for conservation laws: Applications to forward and inverse problems', *Computer Methods in Applied Mechanics and Engineering*, 365: 113028.
Jia, Xiaowei, Jared Willard, Anuj Karpatne, Jordan S Read, Jacob A Zwart, Michael Steinbach, and Vipin %J arXiv preprint arXiv:.11086 Kumar. 2020. 'Physics-guided machine learning for scientific discovery: An application in simulating lake temperature profiles'.
Jidling, Carl, Niklas Wahlström, Adrian Wills, and Thomas Schön. 2017. *Linearly constrained Gaussian processes*.
Jin, Ruichen, Wei Chen, and A. Sudjitanto. 2002. *On Sequential Sampling for Global Metamodeling in Engineering Design*.
Joseph, V. R., and H. Yan. 2015. 'Engineering-Driven Statistical Adjustment and Calibration', *Technometrics*, 57: 257-67.
Joseph, V. Roshan, and Shreyes N. Melkote. 2009. 'Statistical Adjustments to Engineering Models', *Journal of Quality Technology*, 41: 362-75.
Kadeethum, T., Jørgensen, T. M., & Nick, H. M. 2020. 'Physics-informed neural networks for solving nonlinear diffusivity and Biot's equations.', *Plos one*.
Kahrs, O., and W. Marquardt. 2007. 'The validity domain of hybrid models and its application in process optimization', *Chemical Engineering and Processing: Process Intensification*, 46: 1054-66.
———. 2008. 'Incremental identification of hybrid process models', *Computers & Chemical Engineering*, 32: 694-705.
Kalyanaraman, Jayashree, Yanfang Fan, Ying Labreche, Ryan P. Lively, Yoshiaki Kawajiri, and Matthew J. Realff. 2015. 'Bayesian estimation of parametric uncertainties, quantification and reduction using optimal design of experiments for CO2 adsorption on amine sorbents', *Computers & Chemical Engineering*, 81: 376-88.
Kalyanaraman, Jayashree, Yoshiaki Kawajiri, Ryan P. Lively, and Matthew J. Realff. 2016. 'Uncertainty quantification via bayesian inference using sequential monte carlo methods for CO2 adsorption process', *AIChE Journal*, 62: 3352-68.
Karniadakis, Ameya D. Jagtap George Em. 2020a. 'Extended Physics-Informed Neural Networks (XPINNs): A Generalized Space-Time Domain Decomposition Based Deep Learning Framework for Nonlinear Partial Differential Equations', *Communications in Computational Physics*, 28: 2002-41.
Karniadakis, George Em, Ioannis G. Kevrekidis, Lu Lu, Paris Perdikaris, Sifan Wang, and Liu Yang. 2021. 'Physics-informed machine learning', *Nature Reviews Physics*, 3: 422-40.
Karniadakis, Guofei Pang and George Em. 2020b. 'Physics-Informed Learning Machines for Partial Differential Equations: Gaussian Processes Versus Neural Networks', *Nonlinear Systems and Complexity*, 32: 323-43.
Karpatne, A., G. Atluri, J. H. Faghmous, M. Steinbach, A. Banerjee, A. Ganguly, S. Shekhar, N. Samatova, and V. Kumar. 2017. 'Theory-Guided Data Science: A New Paradigm for Scientific Discovery from Data', *IEEE Transactions on Knowledge and Data Engineering*, 29: 2318-31.
Ke, Guolin, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. 2017. "Lightgbm: A highly efficient gradient boosting decision tree." In *Advances in neural information processing systems*, 3146-54.

Kennedy, M. C., and A. O'Hagan. 2001a. 'Bayesian calibration of computer models', *Journal of the Royal Statistical Society Series B-Statistical Methodology*, 63: 425-50.
Kennedy, M., and A. O'Hagan. 2006. "Supplementary details on Bayesian Calibration of Computer Models." In.
Kennedy, Marc C., and Anthony O'Hagan. 2001b. 'Bayesian calibration of computer models', *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 63: 425-64.
Keskitalo, Jukka, and Kauko Leiviskä. 2015. "Artificial Neural Network Ensembles in Hybrid Modelling of Activated Sludge Plant." In *Intelligent Systems'2014*, edited by P. Angelov, K. T. Atanassov, L. Doukovska, M. Hadjiski, V. Jotsov, J. Kacprzyk, N. Kasabov, S. Sotirov, E. Szmidt and S. Zadrożny, 683-94. Cham: Springer International Publishing.
Kevrekidis, Panayotis G, Jesús Cuevas-Maraver, and Avadh Saxena. 2020. *Emerging Frontiers in Nonlinear Science* (Springer).
Kim, Jungeun, Kookjin Lee, Dongeun Lee, Sheo Yon Jin, and Noseong %J arXiv preprint arXiv:.02681 Park. 2020. 'DPM: A Novel Training Method for Physics-Informed Neural Networks in Extrapolation'.
Kim, Young-Jin, and Cheol-Soo Park. 2016. 'Stepwise deterministic and stochastic calibration of an energy simulation model for an existing building', *Energy and Buildings*, 133: 455-68.
Kim, Youngkyu, Youngsoo Choi, David Widemann, and Tarek Zohdi. 2020. 'A fast and accurate physics-informed neural network reduced order model with shallow masked autoencoder', *arXiv preprint arXiv:2009.11990*.
Kissas, G., Yang, Y., Hwuang, E., Witschey, W. R., Detre, J. A., & Perdikaris, P. 2020. 'Machine learning in cardiovascular flows modeling: Predicting arterial blood pressure from non-invasive 4D flow MRI data using physics-informed neural networks.', *Computer Methods in Applied Mechanics and Engineering*.
Krogh, Anders %J Nature biotechnology. 2008. 'What are artificial neural networks?', 26: 195-97.
L'Heureux, A., K. Grolinger, H. F. Elyamany, and M. A. M. Capretz. 2017. 'Machine Learning With Big Data: Challenges and Approaches', *IEEE Access*, 5: 7776-97.
Lagaris, Isaac E, Aristidis Likas, and Dimitrios I Fotiadis. 1998. 'Artificial neural networks for solving ordinary and partial differential equations', *IEEE transactions on neural networks*, 9: 987-1000.
Lange-Hegermann, Markus. 2020. 'Linearly Constrained Gaussian Processes with Boundary Conditions', *ArXiv*, abs/2002.00818.
Lee, D. S., C. O. Jeon, J. M. Park, and K. S. Chang. 2002. 'Hybrid neural network modeling of a full-scale industrial wastewater treatment process', *Biotechnol Bioeng*, 78: 670-82.
Lee, Dae Sung, Peter A. Vanrolleghem, and Jong Moon Park. 2005. 'Parallel hybrid modeling methods for a full-scale cokes wastewater treatment plant', *Journal of Biotechnology*, 115: 317-28.
Lee, Guesuk, Wongon Kim, Hyunseok Oh, Byeng Dong Youn, and Nam Kim. 2019. 'Review of statistical model calibration and validation—from the perspective of uncertainty structures', *Structural and Multidisciplinary Optimization*, 60.
Lee, Jay H., Joohyun Shin, and Matthew J. Realff. 2018. 'Machine learning: Overview of the recent progresses and implications for the process systems engineering field', *Computers & Chemical Engineering*, 114: 111-21.
Lee, Kyungeun, Hyunkyoo Cho, and Ikjin Lee. 2019. 'Variable selection using Gaussian process regression-based metrics for high-dimensional model approximation with limited data', *Structural and Multidisciplinary Optimization*, 59: 1439-54.
Lee, Seungjoon, Felix Dietrich, George E Karniadakis, and Ioannis G Kevrekidis. 2019. 'Linking Gaussian process regression with data-driven manifold embeddings for nonlinear data fusion', *Interface focus*, 9: 20180083.

Li, Kuijun, Priyadarshi Mahapatra, K. Sham Bhat, David C. Miller, and David S. Mebane. 2017. 'Multi-scale modeling of an amine sorbent fluidized bed adsorber with dynamic discrepancy reduced modeling', *Reaction Chemistry & Engineering*, 2: 550-60.

Lin, Li-Hsiang, and V. Joseph. 2019. 'Transformation and Additivity in Gaussian Processes', *Technometrics*, 62: 1-27.

Lindgren, Lars-Erik, Henrik Tersing, and Konstantin Domkin. 2003. 'Constitutive modelling and parameter optimisation'.

Ling, You, Joshua Mullins, and Sankaran Mahadevan. 2014. 'Selection of model discrepancy priors in Bayesian calibration', *Journal of Computational Physics*, 276: 665–80.

Linkletter, Crystal, Derek Bingham, Nicholas Hengartner, David Higdon, and Kenny Q. Ye. 2006. 'Variable Selection for Gaussian Process Models in Computer Experiments', *Technometrics*, 48: 478-90.

Liu, F., M. Bayarri, and J. Berger. 2009. 'Modularization in Bayesian analysis, with emphasis on analysis of computer models', *Bayesian Analysis*, 4.

Liu, Haitao, Yew Ong, Xiaobo Shen, and Jianfei Cai. 2018. *When Gaussian Process Meets Big Data: A Review of Scalable GPs*.

Lopez, Pau Cabaneros, Isuru A. Udugama, Sune T. Thomsen, Christian Roslander, Helena Junicke, Miguel Mauricio-Iglesias, and Krist V. Gernaey. 2020. 'Towards a digital twin: a hybrid data-driven and mechanistic digital shadow to forecast the evolution of lignocellulosic fermentation', *Biofuels, Bioproducts and Biorefining*, 14: 1046-60.

Lu, Junde, Ke Yao, and Furong Gao. 2009. 'Process similarity and developing new process models through migration', 55: 2318-28.

Luo, Linkai, and Furong Gao. 2015. 'Model Migration through Bayesian Adjustments', *IFAC-PapersOnLine*, 48: 112-16.

Lutter, Michael, Christian Ritter, and Jan Peters. 2019. 'Deep lagrangian networks: Using physics as model prior for deep learning', *arXiv preprint arXiv:1907.04490*.

MacKay, David J. C. 2003. *Information theory, inference, and learning algorithms* (Cambridge University Press: Cambridge, UK ; New York).

Manfren, Massimiliano, Niccolò Aste, and Reza Moshksar. 2013. 'Calibration and uncertainty analysis for computer models – A meta-model based approach for integrated building energy simulation', *Applied Energy*, 103: 627-41.

Matsunawa, Tetsuaki, Bei Yu, and David Pan. 2015. *Optical proximity correction with hierarchical Bayes model* (SPIE).

McBride, Kevin, Edgar Ivan Sanchez Medina, and Kai Sundmacher. 2020. 'Hybrid Semi-parametric Modeling in Separation Processes: A Review', *Chemie Ingenieur Technik*, 92: 842-55.

McBride, Kevin, and Kai Sundmacher. 2019. 'Overview of Surrogate Modeling in Chemical Process Engineering', *Chemie Ingenieur Technik*, 91: 228-39.

McCann, Michael T, Kyong Hwan Jin, and Michael %J IEEE Signal Processing Magazine Unser. 2017. 'Convolutional neural networks for inverse problems in imaging: A review', 34: 85-95.

McFarland, John, Sankaran Mahadevan, Vicente Romero, and Laura Swiler. 2008. 'Calibration and Uncertainty Analysis for Computer Simulations with Multivariate Output', *AIAA Journal*, 46: 1253-65.

McIntire, M., D. Ratner, and S. Ermon. 2016. "Sparse Gaussian Processes for Bayesian Optimization." In *UAI*.

Meng, Xuhui, and George Em Karniadakis. 2020. 'A composite neural network that learns from multi-fidelity data: Application to function approximation and inverse PDE problems', *Journal of Computational Physics*, 401: 109020.

Meng, Yanmei, Shuangshuang Yu, Jinlai Zhang, Johnny Qin, Zhen Dong, Guancheng Lu, and Haifeng Pang. 2019. 'Hybrid modeling based on mechanistic and data-driven approaches for cane sugar crystallization', *Journal of Food Engineering*, 257: 44-55.

Misyris, George S, Andreas Venzke, and Spyros %J arXiv preprint arXiv:.03737 Chatzivasileiadis. 2019. 'Physics-Informed Neural Networks for Power Systems'.

Mitusch, Sebastian K., S. Funke, and M. Kuchta. 2021. 'Hybrid FEM-NN models: Combining artificial neural networks with the finite element method', *ArXiv*, abs/2101.00962.

Mohan, Arvind T, Nicholas Lubbers, Daniel Livescu, and Michael Chertkov. 2020. 'Embedding Hard Physical Constraints in Neural Network Coarse-Graining of 3D Turbulence', *arXiv preprint arXiv:2002.00021*.

Narayanan, Harini, Michael Sokolov, Massimo Morbidelli, and Alessandro Butté. 2019. 'A new generation of predictive models: The added value of hybrid models for manufacturing processes of therapeutic proteins', *Biotechnology and Bioengineering*, 0.

Oliveira, R. 2004. 'Combining first principles modelling and artificial neural networks: a general framework', *Computers & Chemical Engineering*, 28: 755-66.

Pakravan, Samira, Pouria A Mistani, Miguel Angel Aragon-Calvo, and Frederic Gibou. 2020. 'Solving inverse-PDE problems with physics-aware neural networks', *arXiv preprint arXiv:2001.03608*.

Palomo, Jesus, Rui Paulo, and Gonzalo Garcia-Donato. 2015. 'SAVE: An R Package for the Statistical Analysis of Computer Models', *Journal of Statistical Software*, 64.

Pan, S. J., and Q. Yang. 2010. 'A Survey on Transfer Learning', *IEEE Transactions on Knowledge and Data Engineering*, 22: 1345-59.

Pang, Guofei, Lu Lu, and George Em Karniadakis. 2019. 'fpinns: Fractional physics-informed neural networks', *SIAM Journal on Scientific Computing*, 41: A2603-A26.

Paszke, Adam, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, and Luca Antiga. 2019. 'Pytorch: An imperative style, high-performance deep learning library', *arXiv preprint arXiv:1912.01703*.

Peherstorfer, Benjamin, Karen Willcox, and Max Gunzburger. 2018. 'Survey of Multifidelity Methods in Uncertainty Propagation, Inference, and Optimization', *SIAM Review*, 60: 550-91.

Perdikaris, Paris, Daniele Venturi, and George Em Karniadakis. 2016. 'Multifidelity information fusion algorithms for high-dimensional systems and massive data sets', *SIAM Journal on Scientific Computing*, 38: B521-B38.

Perdikaris, Paris, Daniele Venturi, Johannes O Royset, and George Em Karniadakis. 2015. 'Multi-fidelity modelling via recursive co-kriging and Gaussian–Markov random fields', *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 471: 20150018.

Piironen, J., and A. Vehtari. 2016. "Projection predictive model selection for Gaussian processes." In *2016 IEEE 26th International Workshop on Machine Learning for Signal Processing (MLSP)*, 1-6.

Pinto, José, Cristiana Rodrigues de Azevedo, Rui Oliveira, and Moritz von Stosch. 2019. 'A bootstrap-aggregated hybrid semi-parametric modeling framework for bioprocess development', *Bioprocess and Biosystems Engineering*, 42: 1853-65.

Plumlee, Matthew. 2017. 'Bayesian Calibration of Inexact Computer Models', *Journal of the American Statistical Association*, 112: 1274-85.

Potharst, R., and A. J. Feelders. 2002. 'Classification trees for problems with monotonicity constraints', 4: 1–10.

Psichogios, Dimitris C., and Lyle H. Ungar. 1992. 'A hybrid neural network-first principles approach to process modeling', *AIChE Journal*, 38: 1499-511.

Qi Zhi He, David Barajas-Solano, Guzel Tartakovsky, Alexandre M. Tartakovksy. 2020. 'Physics-informed neural networks for multiphysics data assimilation with application to subsurface transport', *Advances in Water Resourses*, 141.

Qian, Peter Z. G., and C. F. Jeff Wu. 2008. 'Bayesian Hierarchical Modeling for Integrating Low-Accuracy and High-Accuracy Experiments', *Technometrics*, 50: 192-204.
Qin, S. Joe, and Leo H. Chiang. 2019. 'Advances and opportunities in machine learning for process data analytics', *Computers & Chemical Engineering*, 126: 465-73.
Rackauckas, Christopher, Yingbo Ma, Vaibhav Dixit, Xingjian Guo, Mike Innes, Jarrett Revels, Joakim Nyberg, and Vijay D Ivaturi. 2018. 'A Comparison of Automatic Differentiation and Continuous Sensitivity Analysis for Derivatives of Differential Equation Solutions', *ArXiv*, abs/1812.01892.
Rackauckas, Christopher, Yingbo Ma, Julius Martensen, Collin Warner, Kirill Zubov, Rohit Supekar, Dominic Skinner, and Ali Ramadhan. 2020. "Universal Differential Equations for Scientific Machine Learning." In *arXiv e-prints*, arXiv:2001.04385.
Rackauckas, Christopher, A Singhvi, Y Ma, M Hatherly, SP Jones, C Caine, E Saba, J TagBot, and S Olver. 2020. 'SciML/DifferentialEquations. jl: v6. 15.0'.
Raissi, Maziar. 2019. 'Physics Informed Neural Networks (Github Repository)'.
Raissi, Maziar, Paris Perdikaris, and George E Karniadakis. 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.
Raissi, Maziar, Paris Perdikaris, and George Em Karniadakis. 2017a. 'Inferring solutions of differential equations using noisy multi-fidelity data', *Journal of Computational Physics*, 335: 736-46.
———. 2017b. 'Machine learning of linear differential equations using Gaussian processes', *Journal of Computational Physics*, 348: 683-93.
Rasmussen, C. E., and C. K. I. Williams. 2005. 'Gaussian Processes for Machine Learning', *Gaussian Processes for Machine Learning*: 1-247.
Rasmussen, Carl, and Hannes Nickisch. 2010. 'Gaussian Processes for Machine Learning (GPML) toolbox', *Journal of Machine Learning Research, v.11, 3011-3015 (2010)*, 11.
Rico-Martínez, R., K. Krischer, I. G. Kevrekidis, M. C. Kube, and J. L. Hudson. 1992. 'DISCRETE- vs. CONTINUOUS-TIME NONLINEAR SIGNAL PROCESSING OF Cu ELECTRODISSOLUTION DATA', *Chemical Engineering Communications*, 118: 25-48.
Rüden, Laura von, Sebastian Mayer, Katharina Beckh, B. Georgiev, Sven Giesselbach, R. Heese, Birgit Kirsch, Julius Pfrommer, Annika Pick, Rajkumar Ramamurthy, Michal Walczak, J. Garcke, Christian Bauckhage, and J. Schücker. 2019. 'Informed Machine Learning -- A Taxonomy and Survey of Integrating Knowledge into Learning Systems', *arXiv: Machine Learning*.
Salvatier, John, Thomas V. Wiecki, and Christopher Fonnesbeck. 2016. 'Probabilistic programming in Python using PyMC3', *PeerJ Computer Science*, 2: e55.
Santner, Thomas J., Brian J. Williams, and William Notz. 2003. *The Design and analysis of computer experiments* (Springer: New York).
Sargsyan, K., H. N. Najm, and R. Ghanem. 2015. 'On the Statistical Calibration of Physical Models', *International Journal of Chemical Kinetics*, 47: 246-76.
Särkkä, Simo. 2011. "Linear Operators and Stochastic Partial Differential Equations in Gaussian Process Regression." In, 151-58. Berlin, Heidelberg: Springer Berlin Heidelberg.
Savitsky, Terrance, Marina Vannucci, and Naijun Sha. 2011. 'Variable Selection for Nonparametric Gaussian Process Priors: Models and Computational Strategies', *Statistical science : a review journal of the Institute of Mathematical Statistics*, 26: 130-49.
Schäfer, Pascal, Adrian Caspari, Adel Mhamdi, and Alexander Mitsos. 2019. 'Economic nonlinear model predictive control using hybrid mechanistic data-driven models for optimal operation in real-time electricity markets: In-silico application to air separation processes', *Journal of Process Control*, 84: 171-81.

1738 Schäfer, Pascal, Adrian Caspari, Artur M. Schweidtmann, Yannic Vaupel, Adel Mhamdi, and Alexander
1739 Mitsos. 2020. 'The Potential of Hybrid Mechanistic/Data-Driven Approaches for Reduced
1740 Dynamic Modeling: Application to Distillation Columns', *Chemie Ingenieur Technik*, 92: 1910-20.
1741 Schubert, Jörg, Rimvydas Simutis, Michael Dors, Ivo Havlik, and Andreas Lübbert. 1994. 'Bioprocess
1742 optimization and control: Application of hybrid modelling', *Journal of Biotechnology*, 35: 51-68.
1743 Schulz, Eric, Maarten Speekenbrink, and Andreas Krause. 2018. 'A tutorial on Gaussian process
1744 regression: Modelling, exploring, and exploiting functions', *Journal of Mathematical Psychology*,
1745 85: 1-16.
1746 Schweidtmann, Artur M., Jana M. Weber, Christian Wende, Linus Netze, and Alexander Mitsos. 2021.
1747 'Obey validity limits of data-driven models through topological data analysis and one-class
1748 classification', *Optimization and Engineering*.
1749 Simutis, R., I. Havlik, F. Schneider, M. Dors, and A. Lübbert. 1995. 'ARTIFICIAL NEURAL NETWORKS OF
1750 IMPROVED RELIABILITY FOR INDUSTRIAL PROCESS SUPERVISION.' in A. Munack and K. SchÜGerl
1751 (eds.), *Computer Applications in Biotechnology* (Pergamon: Amsterdam).
1752 Sirignano, Justin, and Konstantinos Spiliopoulos. 2018. 'DGM: A deep learning algorithm for solving
1753 partial differential equations', *Journal of Computational Physics*, 375: 1339-64.
1754 Snelson, Edward, and Zoubin Ghahramani. 2005. *Sparse Gaussian Processes using Pseudo-inputs*.
1755 Snyder, Jon D., and Bala Subramaniam. 1994. 'A novel reverse flow strategy for ethylbenzene
1756 dehydrogenation in a packed-bed reactor', *Chemical Engineering Science*, 49: 5585-601.
1757 Stein, A, and LCA Corsten. 1991. 'Universal kriging and cokriging as a regression procedure', *Biometrics*:
1758 575-87.
1759 Su, Hong-Te, N. Bhat, P. A. Minderman, and T. J. McAvoy. 1992. 'Integrating Neural Networks with First
1760 Principles Models for Dynamic Modeling', *IFAC Proceedings Volumes*, 25: 327-32.
1761 Sun, Bei, Chunhua Yang, Yalin Wang, Weihua Gui, Ian Craig, and Laurentz Olivier. 2020. 'A
1762 comprehensive hybrid first principles/machine learning modeling framework for complex
1763 industrial processes', *Journal of Process Control*, 86: 30-43.
1764 Tagade, Piyush, Krishnan S. Hariharan, Suman Basu, Mohan Kumar Singh Verma, Subramanya Mayya
1765 Kolake, Taewon Song, Dukjin Oh, Taejung Yeo, and Seokgwang Doo. 2016. 'Bayesian calibration
1766 for electrochemical thermal model of lithium-ion cells', *Journal of Power Sources*, 320: 296-309.
1767 Tagade, Piyush M., Byeong-Min Jeong, and Han-Lim Choi. 2013. 'A Gaussian process emulator approach
1768 for rapid contaminant characterization with an integrated multizone-CFD model', *Building and*
1769 *Environment*, 70: 232-44.
1770 Tascikaraoglu, A., and M. Uzunoglu. 2014. 'A review of combined approaches for prediction of short-
1771 term wind speed and power', *Renewable and Sustainable Energy Reviews*, 34: 243-54.
1772 Teixeira, A., A. E. Cunha, J. J. Clemente, J. L. Moreira, H. J. Cruz, P. M. Alves, M. J. T. Carrondo, and R.
1773 Oliveira. 2005. 'Modelling and optimization of a recombinant BHK-21 cultivation process using
1774 hybrid grey-box systems', *Journal of Biotechnology*, 118: 290-303.
1775 Thompson, Michael L., and Mark A. Kramer. 1994. 'Modeling chemical processes using prior knowledge
1776 and neural networks', *AIChE Journal*, 40: 1328-40.
1777 Tipireddy, Ramakrishna, and Alexandre Tartakovsky. 2018. *Physics-informed Machine Learning Method*
1778 *for Forecasting and Uncertainty Quantification of Partially Observed and Unobserved States in*
1779 *Power Grids*.
1780 van Can, H. J. L., H. A. B. te Braake, A. Bijman, C. Hellinga, K. Ch A. M. Luyben, and J. J. Heijnen. 1999. 'An
1781 efficient model development strategy for bioprocesses based on neural networks in
1782 macroscopic balances: Part II', *Biotechnology and Bioengineering*, 62: 666-80.
1783 van Can, H. J. L., H. A. B. te Braake, C. Hellinga, and K. C. A. M. Luyben. 1997. 'An efficient model
1784 development strategy for bioprocesses based on neural networks in macroscopic balances',
1785 *Biotechnology and Bioengineering*, 54: 549-66.

Van Can, Henricus J. L., Chris Hellinga, Karel Ch. A. M. Luyben, Joseph J. Heijnen, and Hubert A. B. Te Braake. 1996. 'Strategy for dynamic process modeling based on neural networks in macroscopic balances', *AIChE Journal*, 42: 3403-18.

Van Can, Henricus J. L., Hubert A. B. Te Braake, Sander Dubbelman, Chris Hellinga, Karel Ch. A. M. Luyben, and Joseph J. Heijnen. 1998. 'Understanding and applying the extrapolation properties of serial gray-box models', *AIChE Journal*, 44: 1071-89.

Venkatasubramanian, Venkat. 2019. 'The promise of artificial intelligence in chemical engineering: Is it here, finally?', *AIChE Journal*, 65: 466-78.

Vincent, Pascal, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. 2008. "Extracting and composing robust features with denoising autoencoders." In *Proceedings of the 25th international conference on Machine learning*, 1096-103.

von Stosch, M., S. Davy, K. Francois, V. Galvanauskas, J. M. Hamelink, A. Luebbert, M. Mayer, R. Oliveira, R. O'Kennedy, P. Rice, and J. Glassey. 2014. 'Hybrid modeling for quality by design and PAT-benefits and challenges of applications in biopharmaceutical industry', *Biotechnol J*, 9: 719-26.

von Stosch, Moritz, Rui Oliveira, Joana Peres, and Sebastião Feyo de Azevedo. 2012. 'A general hybrid semi-parametric process control framework', *Journal of Process Control*, 22: 1171-81.

———. 2014. 'Hybrid semi-parametric modeling in process systems engineering: Past, present and future', *Computers & Chemical Engineering*, 60: 86-101.

Wahlström, N., M. Kok, T. B. Schön, and F. Gustafsson. 2013. "Modeling magnetic fields using Gaussian processes." In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, 3522-26.

Wang, Rui, Robin Walters, and Rose %J arXiv preprint arXiv:.03061 Yu. 2020. 'Incorporating Symmetry into Deep Dynamics Models for Improved Generalization'.

Wang, Sifan, Yujun Teng, and P. Perdikaris. 2020. 'Understanding and mitigating gradient pathologies in physics-informed neural networks', *ArXiv*, abs/2001.04536.

Wang, Xianfang, Jindong Chen, Chunbo Liu, and Feng Pan. 2010. 'Hybrid modeling of penicillin fermentation process based on least square support vector machine', *Chemical Engineering Research and Design*, 88: 415-20.

Wang, Yan, Xiaowei Yue, Rui Tuo, Jeffrey Hunt, and Jianjun Shi. 2019. *Effective Model Calibration via Sensible Variable Identification and Adjustment, with Application to Composite Fuselage Simulation*.

Willard, Jared, Xiaowei Jia, Shaoming Xu, M. Steinbach, and V. Kumar. 2020. 'Integrating Physics-Based Modeling with Machine Learning: A Survey', *ArXiv*, abs/2003.04919.

Willis, Mark J., and Moritz von Stosch. 2017. 'Simultaneous parameter identification and discrimination of the nonparametric structure of hybrid semi-parametric models', *Computers & Chemical Engineering*, 104: 366-76.

Wilson, A., Elad Gilboa, A. Nehorai, and J. Cunningham. 2013. 'GPatt: Fast Multidimensional Pattern Extrapolation with Gaussian Processes', *ArXiv*, abs/1310.5288.

Wilson, Andrew, and Ryan Adams. 2013. 'Gaussian Process Covariance Kernels for Pattern Discovery and Extrapolation', *30th International Conference on Machine Learning, ICML 2013*.

Wilson, Zachary T., and Nikolaos V. Sahinidis. 2017. 'The ALAMO approach to machine learning', *Computers & Chemical Engineering*, 106: 785-95.

Wipf, David, and Srikantan Nagarajan. 2007. *A New View of Automatic Relevance Determination*.

Wu, T., and J. Movellan. 2012. "Semi-parametric Gaussian process for robot system identification." In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 725-31.

Xia, Youshen, Gang Feng, and Jun %J IEEE Transactions on neural networks Wang. 2008. 'A novel recurrent neural network for solving nonlinear optimization problems with inequality constraints', 19: 1340-53.

Xiong, Ying, Wei Chen, Kwok-Leung Tsui, and Daniel W. Apley. 2009. 'A better understanding of model updating strategies in validating engineering models', *Computer Methods in Applied Mechanics and Engineering*, 198: 1327-37.
Yan, F., and Yuan Qi. 2010. "Sparse Gaussian Process Regression via L1 Penalization." In *ICML*.
Yang, Aidong, Elaine Martin, and Julian Morris. 2011. 'Identification of semi-parametric hybrid process models', *Computers & Chemical Engineering*, 35: 63-70.
Yang, Shihao, Samuel W. K. Wong, and S. C. Kou. 2021. 'Inference of dynamic systems from noisy and sparse data via manifold-constrained Gaussian processes', *Proceedings of the National Academy of Sciences*, 118: e2020397118.
Yang, Shu, Pranesh Navarathna, Sambit Ghosh, and B. Wayne Bequette. 2020. 'Hybrid Modeling in the Era of Smart Manufacturing', *Computers & Chemical Engineering*, 140: 106874.
Yang, Xiu, Guzel Tartakovsky, and Alexandre Tartakovsky. 2018. *Physics-Informed Kriging: A Physics-Informed Gaussian Process Regression Method for Data-Model Convergence*.
Yang, Yibo, and Paris Perdikaris. 2018. 'Physics-informed deep generative models', *arXiv preprint arXiv:1812.03511*.
Yi, G., J. Q. Shi, and T. Choi. 2011. 'Penalized Gaussian Process Regression and Classification for High-Dimensional Nonlinear Data', *Biometrics*, 67: 1285-94.
Zendehboudi, Sohrab, Nima Rezaei, and Ali Lohi. 2018. 'Applications of hybrid models in chemical, petroleum, and energy systems: A systematic review', *Applied Energy*, 228: 2539-66.
Zhang, Li, Fulin Wang, Ting Sun, Bing %J Neural Computing Xu, and Applications. 2018. 'A constrained optimization method based on BP neural network', 29: 413-21.

11. Appendix

11.1. Case Study Parameters

The parameter values for base case ethylbenzene reaction and their meaning are listed below:

Additional Information

Catalyst Bed

$\rho = 2137$ [kg/m^3 pellet]

$\phi = 0.4$

Equilibrium Constant

$K_{p1} = exp\left\{b_1 + \frac{b_2}{T} + b_3 \ln(T) + [(b_4 T + b_5)T + b_6]T\right\}$ [atm]

b_1	-17.34	b_4	-2.314 x 10^{-10}
b_2	-1.302 x 10^4	b_5	1.302 x 10^{-6}
b_3	5.051	b_6	-4.931 x 10^{-3}

Heats of Reaction

$\Delta H_{rxn1} = 118$ [kJ/mol ethylbenzene]

$\Delta H_{rxn2} = 105.2$ [kJ/mol ethylbenzene]

$\Delta H_{rxn3} = -53.9$ [kJ/mol ethylbenzene]

Heat Capacities [J/mol/K]

Methane	68	Styrene	27
Ethylene	90	Ethylbenzene	299
Benzene	201	Hydrogen	30
Toluene	249	Steam	40