Generating Diagnosis Models using Integrated Physics-Based/Machine Learning Techniques: A Critical Comparison

Gregory Provan[1]
[1]School of Computer and Information Science, University College Cork, Cork, Ireland
e-mail: g.provan@cs.ucc.ie

Abstract

Combining physics-based approaches and machine learning (ML) to automatically generate models of physical systems is of critical importance, given the limitations of either approach alone. This article empirically compares state-of-the-art parametric and non-parametric methods, based on the availability of moderate amounts of data for a benchmark process-control system. We use a gold-standard ordinary differential equation (ODE) model as a benchmark, provide the learning algorithms with equation fragments and data simulated from the benchmark, and compare the auto-generated models with the benchmark in terms of predictive accuracy, generalizability and equational fidelity. We show that the non-parametric approaches (in particular symbolic regression) generate models that have high test-accuracy, but consist of equations that differ significantly from the benchmark ODEs and have poor generalizability. The parametric approaches that rely heavily on manually-generated model fragments out-perform the non-parametric-based models, but entail greater manual inputs. We discuss the implications of these results on diagnostics modeling efforts for the future.

1 Introduction

A model is at the core of model-based diagnosis (MBD) inference. Traditionally, an MBD model was assumed to be a manually-generated model based on physical principles [1; 2; 3], and if often called a *white-box model*. Machine Learning Models (MLMs), also called (*black-box*) models, have been widely used, typically for applications involving signal-processing [4]. Models that incorporate both physics-based aspects and data-driven methods, i.e., *grey-box models*, as less common for MBD inference, but have been used recently [5; 6; 7]. Grey-box models can make use of data for improving inference, e.g., [8], but still suffer from lack of interpretability of the black-box component.

Manually-generated models based on physical principles are still the *de facto* gold standard for MBD, but this approach is manually intensive, and the models may not incorporate significant information that is not available to and/or known by the persons generating the models. Model identification is widely used to tune models to data [9], but this too has limitations.

MLMs identify multi-dimensional patterns in data, and are agnostic to the underlying physical processes. However, the model-free application of ML has met with limited success in scientific domains due to a number of reasons [10]: (i) while state-of-the-art MLMs can capture complex spatio-temporal relationships, MLMs can have orders of magnitude more free parameters than physics-based models, and hence training requires significant labeled data, which is rarely available in real-world settings; (ii) MLMs often produce scientifically inconsistent results; and (iii) MLMs can only capture relationships in the available training data, and cannot generalize to out-of-sample scenarios (i.e., those not represented in the training data). In addition to these drawbacks, to be able to learn from the available data, MLMs may also require dimensionality reduction and feature engineering, the latter of which is problem-specific [11]. Purely data-drive machine learning thus seems unlikely to bring about high-accuracy diagnostics that are transferable to situations far beyond the available data, without any consideration of physical processes.

Given data $\mathcal{D}$, model generation can be defined as a type of system identification [9]. Common system identification techniques include methods such as (a) classical regression (e.g., using L1, LASSO and L2 or ridge regression), or (b) stepwise regression with statistical tests. These approaches employ non-linear optimization to learn the set of coefficients that result in the best fit to the observations. For adequate data, the system identification problem is relatively robust [9], and together with model-size constraints can be used to learn a parsimonious set of coefficients, especially with stepwise regression.

In general, several other criteria beyond training-set accuracy are necessary for generating MBD models. Criteria that one can trade off during model construction include: accuracy, size, explainability, and model scale. For example, parsimony is central to identifying a well-defined set of equations; the standard strategy to satisfy this requirement is classical or stepwise regression. On another dimension, [12] examines how multi-scale models can be learned.

This article explores the potential of using machine learning (ML) to integrate physical principles with measured data in order to generate white-box MBD models. This approach has been used for automatically generating models in fields such as physics [13], climate modeling [14] and computational biology [15], but has not been adopted widely for MBD. See [16; 17] for recent surveys of this approach.

We compare and contrast several well-known model generation methods, including polynomial regression, Sym-

bolic Regression (SR) and linearization. We use the well-known three-tank system as our benchmark model, from which we generate data $\mathcal{D}$ and automatically learn models.

Our contributions are as follows.

- We use a gold-standard ordinary differential equation (ODE) model as a benchmark, provide the learning algorithms with equation fragments and data simulated from the benchmark, and compare the auto-generated models with the benchmark in terms of predictive accuracy, generalizability and equational fidelity.
- We show that the non-parametric approaches (in particular symbolic regression) generate models that have high test-accuracy, but consist of equations that differ significantly from the benchmark ODEs and have poor generalizability. The parametric approaches that rely heavily on manually-generated model fragments outperform the non-parametric-based models, but entail greater manual inputs.
- We discuss the implications of these results on diagnostics modeling efforts for the future.

2 Preliminaries

This section describes the technical background for our approach.

2.1 Notation

Definition 1 (Symbolic Model)**.** *A symbolic model Φ is a functional representation over a set $\mathcal{X} = \{x_1, ..., x_m\}$ of independent variables and a set $\mathcal{Y} = \{y_1, ..., y_q\}$ of dependent variables, together with parameters θ: $\{y_j = f(x_i, \theta) : x_i \in \mathcal{X},\ y_j \in \mathcal{Y}\}_{j=1,..q}$.*

This definition of model covers many model types, such as ODE, PDE, symbolic (temporal) logic, etc. In particular, we focus on non-linear ODE state-space models Φ, where the first equation is the process model, and the second equation the observation model:

Definition 2 (ODE Physics-based Model)**.** *An ODE PBM is an ODE of the form*

$$\begin{aligned} \dot{x} &= f(x, u) \\ y &= g(x, u), \end{aligned}$$

where the variables x, y have explicit semantic correspondence to physical entities.

There are many issues associated with the notion of level of model, in the terminology of multi-level models. For fluid flow models, one could model at the level of flow regimes (e.g., laminar vs. turbulent), or at a more abstract level where only steady-state conditions are assumed. We do not address those issues in this article.

2.2 Running Example: Three-Tank Benchmark

The system consists of three interconnected tanks, which can have different inflow and outflow stream settings. In this work, we base our analysis based on the system represented in Figure 1. The specific configuration of these settings allow to configure a specific non - linear systems that are normally modeled by PBM approaches. We model this system using a set of ordinary differential equations (ODEs) that represent the dynamics of the content of the fluids within the tanks. Different assumptions over the system (e.g. constant density, no friction, etc.) can be used to derive simpler

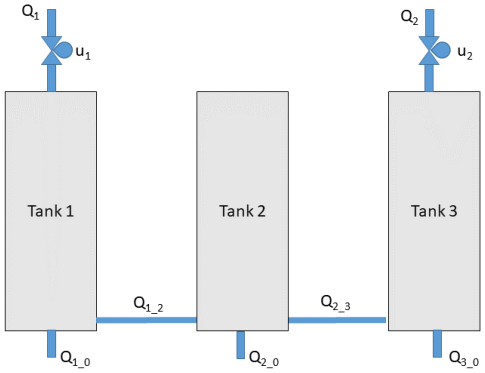

Figure 1: Three tank system.

equations that are useful for simpler system representation, e.g., linear models (see section 4.2).

We represent the model Φ in the state-space form $\dot{\boldsymbol{h}} = f(\boldsymbol{h}, \boldsymbol{u})$, as defined over variables $\boldsymbol{h}, \boldsymbol{u}$, and parameters $g, B, az_{1\_0}, az_{1\_2}, az_{2\_3}$.

$$\frac{dh_1}{dt} = \frac{1}{A}(u_1 Q_1 - Q_{1\_2} - Q_{1\_0}) \quad (1)$$

$$\frac{dh_2}{dt} = \frac{1}{A}(Q_{1\_2} - Q_{2\_3} - Q_{2\_0}) \quad (2)$$

$$\frac{dh_3}{dt} = \frac{1}{A}(u_2 Q_2 + Q_{2\_3} - Q_{3\_0}), \quad (3)$$

where

$$Q_{i\_j} = az_{i\_j} B \sqrt{2g|h_i - h_j|}\sigma(h_i - h_j) \quad (4)$$

$$Q_{i\_0} = az_{i\_0}\sqrt{2gh_i} \quad (5)$$

We use $\sigma(\cdot)$ to denote the sign function of $(\cdot)$. The manipulated variables, u_i, correspond to the valve settings, whose values can be set between 0 and 1.

3 Method

This section describes the methods we use for our experiments.

3.1 System Architecture and Inference

Figure 2 shows the top-level architecture that we adopt for the training phase of our experiments. All experiments make use of a model library, that consists of ODE equations for components, such as a valve, tank, pipe, etc. We use the gold-standard model Φ to generate the data from which we learn the model $\hat{\Phi}$.

Figure 3 shows the top-level architecture that we adopt for the testing phase of our experiments. Here, we compare the performance of the gold-standard model (Φ) and learned model $\hat{\Phi}$.

We performed inference in three main phases, which we detail in the following sections.

1. Simulation-Based Data Generation
2. Model Learning
3. Comparison of Models on Test Data

3.2 Performance Metrics

We evaluated model performance using metrics for model size and accuracy, as well as combined metrics.

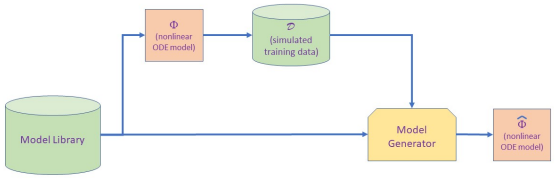

Figure 2: Model generation (training phase). The gold-standard model Φ generates training data from which we learn the model $\hat{\Phi}$

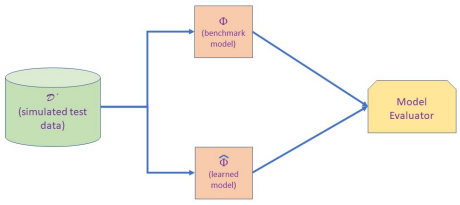

Figure 3: Model analysis (testing phase)

Model Size/Complexity

Measuring model size/complexity is complex, as it entails metrics over the number of variables, model order (e.g., linear vs. higher-order polynomial), and equation complexity. We adopt two measures, which are used widely in evaluating model "quality". First, we use the number n of model variables, which is used for metics like AIC, BIC, etc. Second, we use a measure of equation complexity based on the abstract syntax tree (AST) underlying the equation. For example, Figure 4 shows the AST for the solution to the quadratic formula:

$$x = \frac{b \pm \sqrt{b^2 - 4ac}}{2a}.$$

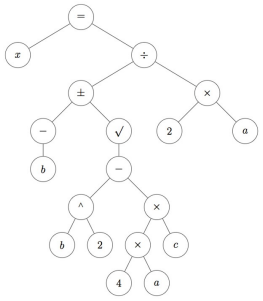

Figure 4: AST for solution to a quadratic formula

We use a measure that captures the number of operators and variables in an AST.

Model Accuracy

We evaluated model performance using several well-known accuracy metrics, which are based on (1) the residuals between the gold-standard and MLM-generated simulation results, and (2) the modeling technique used in the evaluation. We used the Mean Square Error (MSE, Equation 6), Mean Absolute Error (MAE, Equation 7), the Akaike Information Criterion (AIC, Equation 8), an AIC extension (named full, 9) that includes the number of variables considered in the final mathematical expression, and a coefficient of determination dependent value ($1/R^2$). In equations 6 to 10, n_e is the number of points considered in the evaluation, k is the number of predictors, and y_i and $\hat{y}_i$ are the y^{th} real and predicted values of the dependent variables i.

$$MSE_i = = 1/n_e \sum_{i=1}^{n} (\hat{y}_i - y_i)^2 \quad (6)$$

$$MAE_i = = 1/n_e \sum_{i=1}^{n} |\hat{y}_i - y_i| \quad (7)$$

$$AIC_i = n_e + n_e \ln 2\pi + n_e \ln RSS_i/n_e + 2k \quad (8)$$

$$full = AIC_i + n^E_{vars,i} \quad (9)$$

$$R_{base,i} = TSS_i/(TSS_i - RSS_i) \quad (10)$$

We estimate the performance metrics over individual ODEs, and use the agglomerated values (average) for algorithmic purposes (e.g. symbolic regression fitness uses the agglomerated values for individual selection).

4 Automated Model Generation

This section introduces the generic model generation task, and the specific generation algorithms on which we focus.

4.1 Model Generation Task

We define a model generation task as fitting a state-space model Φ to data $\mathcal{D}$.

Definition 3. *Given observational data $\mathcal{D}$ in terms of independent variables, $x_i \in \mathbb{R}^d$, and dependent variables (function values), $z_i \in \mathbb{R}$, for $i = 1, \cdots, n_e$, model generation aims to find a that model Φ that maps the x-values to the z-values by solving the following optimization problem:*

$$\hat{\beta} = \arg\min_{\beta} \frac{1}{2} \parallel z - f(\boldsymbol{x}, \beta) \parallel_2^2 + \alpha\lambda \parallel g(\boldsymbol{x}, \boldsymbol{\beta}) \parallel_1 + (1-\alpha)\lambda \parallel g(\boldsymbol{x}, \boldsymbol{\beta}) \parallel_1,$$

where $0 \leq \lambda$, $\alpha \leq 1$ are regularization parameters, and $g(\boldsymbol{x}, \boldsymbol{\beta})$ is some measure of model complexity.

The first term describes model fitting, and the latter two terms are regularization terms that enable us to control the model complexity. Regularization techniques rely on solving convex continuous optimization problems where the objective is the familiar bias/variance trade-off (between error on the training set, and the magnitude of the regression coefficients). Several loss functions involving general norms are also possible and, in general, Equation 11 denotes an infinite-dimensional optimization problem.

In particular, the parameter α determines the trade-off between the ℓ_1 penalty and the ℓ_2 penalty. If $\alpha = 1$, then equation 11 is equivalent to the lasso problem [18], and will facilitate subset selection as the ℓ_1 penalty will drive regression

coefficients toward zero. If $\alpha = 0$, the problem is equivalent to ridge regression [19]. For any value $0 \leq \alpha \leq 1$, the problem is termed elastic net regression. Although regularization does facilitate subset selection, models generated by Equation 11 are not equivalent to the problem of best subset selection.

In the case of linear models, where the function $f(\boldsymbol{x}, \beta) = \boldsymbol{x}\beta$ and $g(\boldsymbol{x}, \boldsymbol{\beta}) = \beta$, there is a long history of using regularization techniques to address overfitting; even when regression coefficients are driven to zero, model selection is possible [18].

However, unlike traditional system identification, we aim to constrain the process using physics, and there are several approaches that have been adopted. We described the approaches that we adopt in the following sections.

4.2 Parametric Approaches

Model Reduction

Model order reduction (MOR) is the process of taking a complex model whose computational properties preclude efficient inference, and transforming it into a simpler model that trades off some degree of inference accuracy, e.g., [20]. We focus on the process of data-driven model reduction, given data $\mathcal{D}$.

Both parametric [21] and non-parametric [22; 23] methods have been used; here, we employ a parametric approach, as traditionally used in system identification.

Given a gold-standard model $\Phi(\theta)$ and space of possible models $\Gamma = \{\gamma_i(\boldsymbol{\theta_i})\},\ i = 1, \cdots, q$ parameterized by θ_i, MOR seeks to identify the model $\gamma^*(\boldsymbol{\theta^*})$ that trades off accuracy for simplicity, where accuracy is the relative loss $\parallel \mathcal{L}(\Phi) - \mathcal{L}(\gamma^*) \parallel$:

$$\begin{aligned} \gamma^* \quad = \quad & \arg\min_{\gamma_i \in \Gamma} \parallel \mathcal{L}(\Phi) - \mathcal{L}(\gamma^*) \parallel \ + \\ & \alpha\lambda \parallel g(\gamma_i) \parallel_1 \ + \ (1-\alpha)\lambda \parallel g(\gamma_i) \parallel_1 . \end{aligned}$$

Example: Linearization of Three-Tank Model:

We transform the non-linear model $g(\boldsymbol{x},\ \boldsymbol{u}.\theta_{NL})$ into linear state-space form using linearization [**?**]. Here, we fix the form of the model as $\Phi_L(\boldsymbol{x},\ \boldsymbol{u}.\theta_L)$, given by $\dot{\boldsymbol{x}} = A\boldsymbol{x} \ + \ B\boldsymbol{u}$, and we need to find the matrix parameters that best fit to the data, i.e., $\theta_L = [A, B]$:

$$\Phi^* = \arg\min_{\theta} \sum_{d_i \in \mathcal{D}} \frac{1}{2} \parallel g(\boldsymbol{x}, \boldsymbol{u}, \theta_{NL}) - \Phi_L(\boldsymbol{x}, \boldsymbol{u}, \theta_L) \parallel_2^2 .$$

We implement linearization by taking the gradient of the nonlinear function $g(\cdot)$ with respect to all variables and creating a linear representation at an equilibrium point. Consider a nonlinear differential equation model Φ that is derived from balance equations with input u and output y. Given the functional form $\frac{dy}{dt} = f(y, u)$, we linearize the right hand side by a Taylor series expansion, using only the first two terms.

$$\frac{dy}{dt} = f(y,u) \simeq f(\bar{y},\bar{u}) + \frac{\partial f}{\partial y}|_{\bar{y},\bar{u}}(y-\bar{y}) + \frac{\partial f}{\partial u}\ |_{\bar{y},\bar{u}}\ (u-\bar{u}) \tag{11}$$

If the values of $\bar{u}$ and $\bar{y}$ are chosen at steady state conditions then $f(\bar{y}, \bar{u}) = 0$ because the derivative term $\frac{dy}{du} = 0$ at steady state. To simplify the final linearized expression, deviation variables are defined as $y' = y\bar{y}$ and $u' = u\bar{u}$.

We obtain an equation of the form $\dot{\boldsymbol{h}} = A\boldsymbol{h} + B\boldsymbol{u}$, where the full equation is given by

$$\begin{bmatrix} \dot{h_1} \\ \dot{h_2} \\ \dot{h_3} \end{bmatrix} = \begin{bmatrix} -.963 & .954 & 0 \\ .954 & -2.17 & 1.214 \\ 0 & 1.214 & -1.22 \end{bmatrix} \begin{bmatrix} h_1 \\ h_2 \\ h_3 \end{bmatrix} + \begin{bmatrix} .0065 & 0 \\ 0 & 0 \\ 0 & .0065 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

In the experiments, we linearize ODE models using the Matlab linearization toolbox, with an equilibrium point of h_1 = h_2 = h_3 = 0.1 [m] (for the state variables) and u_1 = u_2 = 0.1 (as the control variables). The same initial conditions for the state variables and manipulated variables were used for each simulation.

Even though ODE solvers would report the level of liquid level in the tank systems (i.e. h_1, h_2, and h_3), ODE system identification evaluates the derivative terms (i.e. $C = dh_i/dt$). Derivative values were estimated by estimating $\Delta h/\Delta t$ after ODE-solver calculations.

Multivariate Polynomial Regression (MPR)

Multivariate polynomial regression(MPR) is a regression technique in which the dependant variable is modelled as an n^{th} degree polynomial of the independent variables. MPR is considered to be linear with respect to the unknown parameters, so is considered a special case of linear regressions. We used a Matlab algorithm in our experiments ([24]).

Given observational data in terms of independent variables, $x_i \in \mathbb{R}^d$, and dependent variables (function values), $z_i \in \mathbb{R}$, for $i = 1, \cdots, N_d$, MPR finds the best functional form that maps the x-values to the z-values by solving the following optimization problem:

$$\min_{f \in \mathcal{F}, \beta_i} \sum_{i=1}^{N_d} [\parallel \beta_i f(x_i) - z_i \parallel_2 \ + \ \lambda \parallel g(x_i) \parallel], \tag{12}$$

where $\mathcal{F}$ is the space of n^{th} degree polynomial functions from which f is chosen, β_i is the parameter for $f(x_i)$, $0 \leq \lambda \leq 1$ is a regularization parameter, and $g(x_i)$ is some measure of model complexity.

Variable Fidelity Model Generation

Variable-fidelity modeling (VFM) [25; 26; 27] can be viewed as a model-generation task in which we generate an ensemble of k (simple) models, weighted by parameters $\beta_i,\ i = 1, ..., k$, to approximate a gold-standard model $\Phi(\theta)$. Similar to MOR, VFM aims to select from a space of possible models $\Gamma = \{\gamma_i(\boldsymbol{\theta_i})\},\ i = 1, \cdots, q$ parameterized by θ_i, a composite model $\gamma^*(\boldsymbol{\theta^*}) = \sum_i \beta_i \gamma_i(\theta_i)$ that optimizes accuracy with respect to Φ, where accuracy is the relative loss $\parallel \mathcal{L}(\Phi) - \mathcal{L}(\gamma^*) \parallel$:

$$\arg\min_{\beta, \gamma_i \in \Gamma} \parallel \mathcal{L}(\sum_i \beta_i \gamma_i) - \mathcal{L}(\Phi) \parallel \tag{13}$$

We represent a system $\mathcal{S}$ using a high-fidelity model (HFM), Φ. We assume that the system $\mathcal{S}$ can also be simulated with lower-fidelity models (LFMs), each of which represents a simplification of the HFM. We represent a LFM using Φ_L: $\boldsymbol{y}_L = \psi(\boldsymbol{x}_L)$. We also assume that the HFM takes into account a larger number of variables, parameters and processes describing the physical system $\mathcal{S}$ than does

any LFM; hence, inference using the HFM is likely to be more computationally demanding than that with the LFMs. More precisely, our assumption entails that the state vector of a lower-fidelity model, $\boldsymbol{x}_L$, is a subset of the state vector of the HFM: $\boldsymbol{x}_L \subseteq \boldsymbol{x}_H$. We represent a collection of lower-fidelity models using $\mathcal{Y} = \{\phi_{L^1}, \cdots, \Phi_{L^k}\}$.

In the general case, we model the relationship between the outputs from the HFM and from the lower-fidelity models using a mathematical function ξ, such that the output from the HFM can be written as:

$$\boldsymbol{y}_H = \xi(\mathcal{Y}, \beta) + \epsilon \tag{14}$$

where ξ is a mathematical function (denoted as a "linking function"), $\boldsymbol{\beta}$ is a vector of unknown parameters of the linking function, and ϵ is an error term. Equation 14 uses the linking function as a surrogate model to "link" outputs from models with different levels of fidelity. If we treat $\xi(\mathcal{Y}, \beta)$ as a random process, then we can describe this task in terms of a Gaussian Process [26]. In this article, we restrict our VFM models to linear combinations, but more general forms are possible.

4.3 Non-Parametric Approach: Symbolic Regression

We use symbolic regression (SR) for our parametric approach since it has been shown to outperform other parametric models for small data sets [28]. Unlike traditional regression, symbolic regression (SR) does not assume a fixed functional form; instead, SR learns the functional relationship and its constants [29]. Symbolic regression subsumes linear regression, generalized linear regression, and generalized additive models into a larger class of methods. Symbolic regression minimizes a given loss function defined over the functional form of a regression function and the associated parameters or coefficients. The functional form is assumed to be anything that can be composed from a given list of basic functions or operators applied to the independent variables and arbitrary constants. For example, if the operators are $+$, and $\times$, then the space of all possible functions is the set of all polynomials of arbitrary degree.

A symbolic regression algorithm conducts a search over the space of possible model data structures that represent valid mathematical expressions. The *data structure* is an AST: a rooted binary tree where each non-leaf node has an associated binary or unary operator ($+, , \times, \sqrt{\cdot}, \log$, etc.), and each leaf node has an associated constant or independent variable. We also use operators denoting the absolute-value $|\cdot|$ and sign $\sigma(\cdot)$. The *search algorithm* is typically chosen to be a genetic algorithm, although several other search methods are applicable.

Given data $\mathcal{D}$, symbolic regression solves the following optimization problem:

$$\min_{f \in \mathcal{F}} \sum_{i=1}^{N_d} [\parallel f(x_i) - z_i \parallel \; + \; \lambda \parallel g(x_i) \parallel, \tag{15}$$

where $\mathcal{F}$ is the space of functions from which f is chosen, $0 \leq \lambda \leq 1$ is a regularization parameter, and $g(x_i)$ is some measure of model complexity.

In this article, we use both physics-free and physics-based kernels to populate $\mathcal{F}$, in order to compare the impact of the different kernels. Table 1 shows some of the kernels that we have studied in equation generation. Kernel K_1 is a kernel consisting only of mathematical operators (physics-free), while the other kernels contain physical model equations. The equation fragments, which are derived from Equation 4, are defined in Table 2.

Kernel	Description
K_1	$\times, /, +, -, \lvert\cdot\rvert, \sqrt{\nu}, \nu^2, \sigma(\cdot), \sin, \cos, e^{\nu}, \log(\nu)$
K_2	$K_1 \cup \mathcal{E}_1$
K_3	$K_1 \cup \mathcal{E}_2$
K_4	$K_1 \cup \mathcal{E}_3$

Table 1: Symbolic regression kernels used in equation generation, described using arbitrary variable ν for K_1 and physical model fragments $\mathcal{E}_1, \mathcal{E}_2, \mathcal{E}_3$

Name	Equations
$\mathcal{E}_1$	$\frac{1}{A}(Q_1 x_1 - a z_{1\_2} S \sigma(h_1 - h_2)(2g\lvert h_1 - h_2\rvert)^{0.5})$
	$\frac{1}{A}(a z_{1\_2} \sigma(h_1 - h_2)(2g\lvert h_1 - h_2\rvert)^{0.5})$
$\mathcal{E}_2$	$\frac{1}{A}(Q_1 x_1 - a z_{1\_2} S \sigma(h_1 - h_2)(2g\lvert h_1 - h_2\rvert)^{0.5})$
$\mathcal{E}_3$	$\frac{1}{A}(Q_1 x_1 - a z_{1\_2} S \sigma(h_1 - h_2)(2g\lvert h_1 - h_2\rvert)^{0.5})$
	$\frac{1}{A}(a z_{1\_2} S \sigma(h_1 - h_2)(2g\lvert h_1 - h_2\rvert)^{0.5})$
	$\frac{1}{A}(a z_{2\_3} S \sigma(h_2 - h_3)(2g\lvert h_2 - h_3\rvert)^{0.5})$

Table 2: Physical model fragments $\mathcal{E}_1, \mathcal{E}_2, \mathcal{E}_3$ used in symbolic regression experiments

We define for the size function $g(x)$ the size χ of the AST of the SR expression, by summing the number of operators and symbols in the expression.

5 Diagnostic Evaluation

This section evaluates the ability of the generated models to compute diagnoses for the benchmark model.

5.1 Diagnosis Model and Fault Simulation

We create a diagnosis model Φ_D from a "simulation" model Φ by adding fault parameters and fault behaviours to Φ. We define a set $\boldsymbol{\varphi}$ of failure parameters to the model, as described in Table 3. To define abnormal failure behaviours, we add an equation describing the failure behaviour for each $\varphi_i, i = 1, \cdots, 7$. Equations 16 through 20 show the fault model, with the fault parameters shown in red. For example, an actuator fault corresponding to to inflow u_1 is parameterized by φ_1, which when set to a value below 1 indicates a sub-normal input flow denoting the actuator is stuck partially open.

$$\frac{dh_1}{dt} = \frac{1}{A}(\varphi_1 u_1 Q_1 - Q_{1\_2} - Q_{1\_0}) \tag{16}$$

$$\frac{dh_2}{dt} = \frac{1}{A}(Q_{1\_2} - Q_{2\_3} - Q_{2\_0}) \tag{17}$$

$$\frac{dh_3}{dt} = \frac{1}{A}(\varphi_2 u_2 Q_2 + Q_{2\_3} - Q_{3\_0}), \tag{18}$$

where

$$Q_{i\_j} = \varphi_{j+1} a z_{i\_j} B \sqrt{2g\lvert h_i - h_j\rvert} \sigma(h_i - h_j) \tag{19}$$

$$Q_{i\_0} = \varphi_{i+5} a z_{i\_0} S n \sqrt{2gh_i}. \tag{20}$$

Fault Type	Fault Parameter	Component	fault range
Actuator	φ_1	u_1	$[0-0.8]$
	φ_2	u_2	$[0-0.8]$
pipe	φ_3	P_{12}	$[0-0.8]$
	φ_4	P_{23}	$[0-0.8]$
leak	φ_5	T_1	$[0-0.8]$
	φ_6	T_2	$[0-0.8]$
	φ_7	T_3	$[0-0.8]$

Table 3: Fault parameters for diagnostics evaluation

Type	**Model**	**Description**	R^2
Parametric	Linearized	4.2	0.998
	MPR	4.2	0.9996
	VFM	4.2	0.99
Non-Parametric	SR	4.3	0.998

Table 4: Models adopted for experimental evaluation

We simulated data, denoted $\mathcal{D}_\delta$, using the diagnosis version of the gold-standard model Φ_D, and then used the diagnosis versions of the learned models for fault isolation. Table 3 describes the fault scenarios that we used, comprising single-faults to (1) the actuators governing inflow of fluid; (2) pipes connecting tanks 1 to 2 and tanks 2 to 3; and (3) leaks in tanks $T_1,\ T_2,\ T_3$.

5.2 Experimental Design

We automatically generated four classes of models: 3 parametric models (linearized, MPR and VFM), and several versions of the non-parametric SR-derived model. For our comparative analysis, we selected learned models that had high R^2 values on the training data. The MLMs adopted are: (1) linearized model (as described in Section 4.2); (2) Multivariate Polynomial Regression (MPR) model (as described in Section 4.2); (3) variable fidelity model (VFM) (as described in Section 4.2); (4) non-parametric. Table 4 summarizes the models we have generated.

For each scenario, we compute a discrete probability distribution $P(\boldsymbol{\varphi})$ over the discrete set of faults $\boldsymbol{\varphi}$. We use for our diagnosis correctness metric the KL divergence over the true and computed distributions, denoted $P^*(\boldsymbol{\varphi})$ and $Q(\boldsymbol{\varphi})$, respectively. We define this metric as follows:

Definition 4 (KL divergence)**.** *For discrete probability distributions P^* and Q defined on the same probability space, $\mathcal{V}$, the relative entropy from Q to P^* is defined to be $D_{KL}(P^* \parallel Q) = \sum_{v\in\mathcal{V}} P^*(v)\log\left(\frac{P^*(v)}{Q(v)}\right)$. Here $Q(v)$ is the approximation and $P^*(v)$ is the gold-standard distribution we're interested in matching $Q(v)$ to.*[1]

5.3 Experimental Results

We learned a range of models for each of the parametric and non-parametric classes. As an example of the output, Table 6 shows a selection of SR models generated for the first 10 of the scenarios we examined. Table 7 shows a two MPR models generated for scenarios 32-33; no kernels are used for this algorithm.

[1]Intuitively this measures the how much a given arbitrary distribution is away from the true distribution. If two distributions perfectly match, $D_{\text{KL}}(P^* \parallel Q) = 0$ otherwise it can take values between 0 and ∞. The lower the KL divergence value, the better we have matched the true distribution P^* with Q.

Model	nominal	actuator fault	pipe blockage fault	leak fault
Φ_{GS}	0.001	0.003	0.006	0.01
Linearized	0.08	0.09	0.16	0.25
VFM	0.07	0.08	0.11	0.18
MPR	0.24	21.7	27.5	31.2
$SR_\emptyset$	0.31	87.2	101.6	129.9
SR_K	0.21	76.1	82.1	103.4

Table 5: Experimental results for diagnostics evaluation

For each model generation method, we selected one of the best-performing output models to perform diagnostics evaluation. We ran experiments for each of the failure-mode types, and averaged the KL-divergence over that type. Table 5 summarizes the results obtained, where the smaller the value the better the performance. In the table, $SR_\emptyset$ and SR_K denote the SR equations generated using kernels K_1 (no physics-based inputs), and K_2 or K_3 (physics-based inputs), respectively. We include diagnostics results from the gold standard model Φ_{GS} as a baseline for the other models, since no diagnostics algorithms will perform perfectly.

Table 5 shows that the parametrics approaches that use significant model information perform relatively well. The linearized model uses a fixed model structure that incorporates all critical variables from the ODE model Φ_{GS}. The VFM model corresponds to a linear combination of (simplified) physics-based models, and its good performance mimics the excellent performance demonstrated in ML for ensembles of simple classifiers [30].

Table 5 shows that the MPR and the SR (non-parametrics) approaches perform significantly worse than the parametrics approaches that use significant model information. Unlike the linearized model, these approaches are not automatically constrained to incorporate all critical variables from the ODE model, and the lack of control variables in some of these models hinders their performance, among other reasons.

Also note that the generated equations (Tables 6 and 7) show that the generated equations differ significantly from the gold standard equations in Φ_{GS}. Hence although the R^2 performance measure on the training data is excellent, the generalization performance for diagnostics is quite poor.

6 Summary and Discussion

6.1 Discussion of Results

This article has presented a critical comparison of parametric and non-parametric approaches to automated generation of diagnostics models. The results indicate that the greater the degree of physics-based information, the better the performance of the resulting diagnostics models. In particular, the VFM approach performed best, in that it uses an ensemble of pre-defined physics-based models. We can call the VFM and linearized methods *strong* physics-based models.

A second important outcome is that the approaches that attempted to learn significant model information performed much worse than the strong physics-based models. This can be attributed to many reasons.

Sample Complexity: We have explicitly investigated methods that work with limited data. Even given the rep-

#	Metric	Kernel	Equation generated	R^2	χ
1	full	K_1	$4.776e^{-6}x_1 - 9.552e^{-6}x_2 - 4.776e^{-6}\sigma(x_2) + 4.776e^{-6}x_1^{1/2} + 4.776e^{-6}x_1\sigma(x_1 - x_2) + 3.992e^{-5}$	0.9902	38
2	full	K_1	$1.192e^{-5}(x_1 - x_2) + 2.383e^{-5}\sigma(x_1 - x_2) + 1.192e^{-5}\sigma(x_1) - 6.984e^{-6}$	0.9817	33
3	AIC	K_1	$1.214e^{-5}x_1 - 1.214e^{-5}x_2 + 2.427e^{-5}\sigma(x_1 - x_2) + 1.214e^{-5}\sigma(x_1) - 1.214e^{-5}x_1^{1/2} + 2.836e^{-5}$	0.9856	41
4	AIC	K_1	$1.307e^{-5}x_1 - 1.307e^{-5}x_2 + 1.307e^{-5}\sigma(x_1 - x_2) + 1.307e^{-5}\sin(\lvert(x_1)^{1/2}\rvert) + 1.307e^{-5} * \sigma(x_1) - 6.771e^{-6}$	0.98264	36
5	MSE	K_1	$1.148e^{-5}x_1 - 1.148e^{-5}x_2 + 1.148e^{-5}\lvert cos(x_2)\rvert - 1.148e^{-5}\cos(x_2) + 1.148e^{-5}\sigma(x_1) - 1.623e^{-5}x_2^{1/2} + 3.727e^{-5}$	0.97614	40
6	MSE	K_1	$4.87e^{-5}\sigma(h_1 - h_2)\lvert h_1 - h_2\rvert^{0.5} - 8.131e^{-20}$	1	19
7	MSE	K_1	$1.214e^{-5}x_1 - 1.214e^{-5}x_2 - 1.214e^{-5}x_2^{1/2} + 4.015e^{-5}$	0.96865	19
8	MSE	K_1	$4.87e^{-5}\sigma(h_1 - h_2)\lvert h_1 - h_2\rvert^{0.5} - 8.131e^{-20}$	1	19
9	MSE	K_2	$4.558e^{-5}\sigma(x_2) - 4.558e^{-5}\sigma(x_1) + 4.558e^{-5}x_2^{1/4}\sigma(x_1 - x_2) + 4.558e^{-5}x_1^{1/2} - 4.558e^{-5}x_2^{1/2} + 1.116e^{-6}$	0.9961	48
10	MSE	K_2	$4.87e^{-5}\sigma(h_1 - h_2)\lvert h_1 - h_2\rvert^{0.5} - 8.131e^{-20}$	1	19

Table 6: Symbolic Regression results for scenarios 1-10, with equations generated given R^2 value and equation complexity χ

#	Metric	Equation generated	R^2	χ
52	MSE	$548.09h_3 - 185.22h_2 + 1777.945h_2h_3 + 567.359h_1 - 15428.530h_1h_3 + -239771.9661h_1h_2 + 8.1236 + 124766.4943h_1^2 + 119902.5816h_2^2 + 9600.560h_3^2$	0.9996	27
33	MSE	$0.0011\lvert x_1 + x_2 - 2x_3\rvert/(\lvert\lvert x_1 - u_1 + x_1^2\rvert - \lvert u_2 + x_2 - x_3^{0.5}\rvert\rvert) - 0.0060u_1 + 0.0011\lvert x_1 - h_2\rvert^{0.5}/\lvert[\lvert h_1 - u_1 + x_1^2\rvert - \lvert u_2 + x_2 - x_3^{0.5}\rvert]\rvert + 0.0029\sigma(x_1 - x_2)\lvert x_1 - x_2\rvert^{0.5} - 5.9760e^{-4}$	0.9999	68

Table 7: MPR results for scenarios 32-33, with equations generated given R^2 value and equation complexity χ

utation for SR of generating good results with limited data [28], our study indicates that for complex models (even for a toy model like the 3-tank system), significantly more data is required.

Data Adequacy: To date, non-parametric methods like SR and deep learning have had success on very simple, constrained model fragments that correspond to trivial physical models. In this process-control domain, such simple physical models may correspond to components like the valve, actuator, pipe, etc. The data used is not of fine enough granularity to enable non-parametric methods to learn good models. However, in process control domains with faults, it is unlikely that data of that nature is readily available, let alone sufficient failure data, to meet such data adequacy requirements.

Structure-Based/Causality Information: It appears as if using model structure could also play a big role in improving the performance of learning. This may also be related to causal information, e.g., abnormal flows are *caused* by upstream pipe-blockage faults. In this example, structure (upstream) is linked with fault causation. In these experiments no information of this nature was used, beyond the actual physical equations.

6.2 Next Steps

This work represents a preliminary study of model learning incorporating physics. By taking a different approach to most prior work, i.e., studying the ability to generate industrial-type models with typical industrial limitations of data, we have opened a major avenue for future work. An open question is whether non-parametric methods are in fact limited to component-scale learning, in which case the methods like VFM/linearization would play a big role into the future. Incorporating notions of model structure and causality also could be significant, especially for diagnostics applications. We plan to examine such questions in future work.

References

[1] Peter Struss. Towards a theory of modeling for diagnosis. *Readings in model-based diagnosis*, page 419, 1992.

[2] Peter Lucas. Symbolic diagnosis and its formalisation. *The Knowledge Engineering Review*, 12(2):109–146, 1997.

[3] Edi Muškardin, Ingo Pill, and Franz Wotawa. Catio-a framework for model-based diagnosis of cyber-physical systems. In *International Symposium on Methodologies for Intelligent Systems*, pages 267–276. Springer, 2020.

[4] S Joe Qin. Survey on data-driven industrial process monitoring and diagnosis. *Annual reviews in control*, 36(2):220–234, 2012.

[5] Jiequn Han, Linfeng Zhang, et al. Integrating machine learning with physics-based modeling. *arXiv preprint arXiv:2006.02619*, 2020.

[6] Belarmino Pulido, Jesús M Zamarreño, Alejandro Merino, and Anibal Bregon. State space neural networks and model-decomposition methods for fault diagnosis of complex industrial systems. *Engineering Applications of Artificial Intelligence*, 79:67–86, 2019.

[7] Daniel Jung. Residual generation using physically-based grey-box recurrent neural networks for engine fault diagnosis. *arXiv preprint arXiv:2008.04644*, 2020.

[8] Ando Andriamamonjy, Ralf Klein, and Dirk Saelens. Automated grey box model implementation using bim and modelica. *Energy and Buildings*, 188:209–225, 2019.

[9] L. Ljung. *System identification – theory for the user*. Prentice-Hall, 2nd edition, 1999.

[10] Anuj Karpatne, Gowtham Atluri, James H Faghmous, Michael Steinbach, Arindam Banerjee, Auroop Ganguly, Shashi Shekhar, Nagiza Samatova, and Vipin Kumar. Theory-guided data science: A new paradigm for scientific discovery from data. *IEEE Transactions on knowledge and data engineering*, 29(10):2318–2331, 2017.

[11] Kristen A Severson, Peter M Attia, Norman Jin, Nicholas Perkins, Benben Jiang, Zi Yang, Michael H Chen, Muratahan Aykol, Patrick K Herring, Dimitrios Fraggedakis, et al. Data-driven prediction of battery cycle life before capacity degradation. *Nature Energy*, 4(5):383–391, 2019.

[12] Grace CY Peng, Mark Alber, Adrian Buganza Tepole, William R Cannon, Suvranu De, Savador Dura-Bernal, Krishna Garikipati, George Karniadakis, William W Lytton, Paris Perdikaris, et al. Multiscale modeling meets machine learning: What can we learn? *Archives of Computational Methods in Engineering*, 28(3):1017–1037, 2021.

[13] Elizabeth Qian, Boris Kramer, Benjamin Peherstorfer, and Karen Willcox. Lift & learn: Physics-informed machine learning for large-scale nonlinear dynamical systems. *Physica D: Nonlinear Phenomena*, 406:132401, 2020.

[14] K Kashinath, M Mustafa, A Albert, JL Wu, C Jiang, S Esmaeilzadeh, K Azizzadenesheli, R Wang, A Chattopadhyay, A Singh, et al. Physics-informed machine learning: case studies for weather and climate modelling. *Philosophical Transactions of the Royal Society A*, 379(2194):20200093, 2021.

[15] Chris Lawson, Jose Manuel Martí, Tijana Radivojevic, Sai Vamshi R Jonnalagadda, Reinhard Gentz, Nathan J Hillson, Sean Peisert, Joonhoon Kim, Blake A Simmons, Christopher J Petzold, et al. Machine learning for metabolic engineering: A review. *Metabolic Engineering*, 2020.

[16] Jared Willard, Xiaowei Jia, Shaoming Xu, Michael Steinbach, and Vipin Kumar. Integrating physics-based modeling with machine learning: A survey. *arXiv preprint arXiv:2003.04919*, 2020.

[17] Donal P Finegan, Juner Zhu, Xuning Feng, Matt Keyser, Marcus Ulmefors, Wei Li, Martin Z Bazant, and Samuel J Cooper. The application of data-driven methods and physics-based learning for improving battery safety. *Joule*, 2020.

[18] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288, 1996.

[19] Arthur E Hoerl and Robert W Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67, 1970.

[20] Renee Swischuk, Laura Mainini, Benjamin Peherstorfer, and Karen Willcox. Projection-based model reduction: Formulations for physics-based machine learning. *Computers & Fluids*, 179:704–717, 2019.

[21] Peter Benner, Serkan Gugercin, and Karen Willcox. A survey of projection-based model reduction methods for parametric dynamical systems. *SIAM review*, 57(4):483–531, 2015.

[22] Stefania Fresca, Andrea Manzoni, et al. A comprehensive deep learning-based approach to reduced order modeling of nonlinear time-dependent parametrized pdes. *Journal of Scientific Computing*, 87(2):1–36, 2021.

[23] Han Gao, Jian-Xun Wang, and Matthew J Zahr. Non-intrusive model reduction of large-scale, nonlinear dynamical systems using deep learning. *Physica D: Nonlinear Phenomena*, 412:132614, 2020.

[24] Ahmet Cecen. Multivariate Polynomial Regression.

[25] Ping Jiang, Qi Zhou, and Xinyu Shao. *Surrogate Model-Based Engineering Design and Optimization*. Springer, 2020.

[26] Jack PC Kleijnen. Kriging metamodeling in simulation: A review. *European journal of operational research*, 192(3):707–716, 2009.

[27] Nicolas Courrier, Pierre-Alain Boucard, and Bruno Soulier. Variable-fidelity modeling of structural analysis of assemblies. *Journal of Global Optimization*, 64(3):577–613, 2016.

[28] Casper Wilstrup and Jaan Kasak. Symbolic regression outperforms other models for small data sets. *arXiv preprint arXiv:2103.15147*, 2021.

[29] Ivan Zelinka, Zuzana Oplatkova, and Lars Nolle. Analytic programming–symbolic regression by means of arbitrary evolutionary algorithms. *International Journal of Simulation: Systems, Science and Technology*, 6(9):44–56, 2005.

[30] Xibin Dong, Zhiwen Yu, Wenming Cao, Yifan Shi, and Qianli Ma. A survey on ensemble learning. *Frontiers of Computer Science*, 14(2):241–258, 2020.