Received January 19, 2021, accepted January 25, 2021, date of publication January 28, 2021, date of current version February 8, 2021.

*Digital Object Identifier 10.1109/ACCESS.2021.3055353*

# Incorporating Unmodeled Dynamics Into First-Principles Models Through Machine Learning

**WARD QUAGHEBEUR[1,2], INGMAR NOPENS[2], AND BERNARD DE BAETS[1]**
[1]KERMIT, Department of Data Analysis and Mathematical Modeling, Ghent University, 9000 Gent, Belgium
[2]BIOMATH, Department of Data Analysis and Mathematical Modeling, Ghent University, 9000 Gent, Belgium

Corresponding author: Ward Quaghebeur (ward.quaghebeur@ugent.be)

This work was supported in part by the Research Foundation - Flanders (FWO) through under Grant 3S79219, and in part by the Flemish Government under the ''Onderzoeksprogramma Artificiële Intelligentie (AI) Vlaanderen'' Programme.

**ABSTRACT** First-principles modeling of dynamical systems is a cornerstone of science and engineering and has enabled rapid development and improvement of key technologies such as chemical reactors, electrical circuits, and communication networks. In various disciplines, scientists structure the available domain knowledge into a system of differential equations. When designed, calibrated, and validated appropriately, these equations are used to analyze and predict the dynamics of the system. However, perfect knowledge is usually not accessible in real-world problems. The incorporated knowledge thus is a simplification of the real system and is limited by the underlying assumptions. This limits the extent to which the model reflects reality. The resulting lack of predictive power severely hampers the application potential of such models. Here we introduce a framework that incorporates machine learning into existing first-principles modeling. The machine learning model fills in the knowledge gaps of the first-principles model, capturing the unmodeled dynamics and thus improving the representativeness of the model. Moreover, we show that this approach lowers the data requirements, both in quantity and quality, and improves the generalization ability in comparison with a purely data-driven approach. This approach can be applied to any first-principles model with sufficient data available and has tremendous potential in many fields.

**INDEX TERMS** Differential equations, dynamical systems, first-principles modeling, hybrid modeling, machine learning.

Differential equations are ubiquitous for modeling dynamical systems based on first-principles [1], [2]. In physics, chemistry, biology, and other fields, scientists try to structure domain knowledge in a system of differential equations [3]. The parameters of these equations are subsequently estimated from relatively few observations. The resulting model can then be integrated by a numerical solver to reproduce the dynamics of the system and make predictions. As the estimated parameters are limited in number and have a physical meaning, the model is interpretable and can improve understanding of the system. However, the available knowledge of the system is usually not perfect, as it is incomplete, a simplification, and subject to certain assumptions. This uncertainty of the model structure propagates and worsens the extent to which the model reflects the real system.

When domain knowledge of the dynamical system is lacking, the model structure can be learned from data. Symbolic regression attempts to find the model structure in the space of mathematical expressions to find the model that best fits the dataset, balancing complexity and accuracy [4]–[6]. Symbolic regression, usually performed through genetic programming [7], [8], can potentially retrieve any structure, but is computationally expensive and scales poorly to larger systems. Alternatively, the underlying structure can be discovered through sparse regression over a library of candidate functions, often polynomials [9], [10]. This approach intrinsically balances complexity with accuracy in a computationally efficient manner, but is obviously limited by the candidate functions present in its library. When the underlying

The associate editor coordinating the review of this manuscript and approving it for publication was Kaustubh Raosaheb Patil.

structure cannot be expressed as a sum of these functions, sparse regression fails.

Other techniques can be used to learn the structure from the available data in a black-box manner, such as radial basis functions [11], Gaussian processes [12], [13], equation-free models [14], or nonlinear autoregressive models [15]. Here, we use neural networks [16]–[19], as these can approximate any arbitrary function [20], [21] and can be trained through gradient descent. This concept is known under various names such as continuous-time recurrent neural networks [22], [23] and dynamic neural networks [24], but is currently mostly known under the name of neural differential equations [25].These neural differential equations can approximate any underlying dynamics and thereby make highly accurate predictions. However, their approximation capabilities make them prone to overfitting to noise. In comparison to first-principles models, neural differential equations need a large and representative dataset and therefore fail to extrapolate to regions not seen before. Furthermore, as their parameters have no physical meaning, they lack interpretability. Similarly, data-driven discovery of nonlinear partial differential equations, incorporating a spatial component, can be done with Physics Informed Neural Networks [26].

In reality, knowledge of the system is often available to a certain extent. This domain knowledge has been constructed from theory and empirical knowledge of the system but is often a simplification, incomplete, and subject to certain assumptions. Here, we introduce a framework that incorporates both (incomplete) knowledge and data into a hybrid model. Dynamics that are not modeled explicitly by the first-principles component are captured by the machine learning component, thereby filling in knowledge gaps. This improves the representativeness of the model. From a machine learning perspective, incorporating domain knowledge has a certain regularizing effect that improves the generalization ability of the model and lowers the data requirements, both in quantity and quality.

## I. METHOD

We consider a first-principles model consisting of a system of differential equations of the form

$$\frac{d^k\mathbf{X}(t)}{dt^k} = f(\mathbf{X}(t); \mathbf{p}), \tag{1}$$

where $\mathbf{X}(t) \in \mathbb{R}^n$ is an $n$-dimensional vector representing the state at time $t$ and $f(X(t); \mathbf{p}) : \mathbb{R} \rightarrow \mathbb{R}^n$ is function parameterized by vector $\mathbf{p}$ capturing the dynamics of the system. This form can be extended straightforwardly to include time dependency, forcing, or spatial dynamics. In first-principles models, this right-hand side would be constructed from theory and empirical knowledge, e.g., mass balances or measured reaction rates.

In a neural differential equation, the structure is not predefined but rather discovered from data. We consider a neural differential equation model of the form

$$\frac{d^k\mathbf{X}(t)}{dt^k} = n(\mathbf{X}(t); \mathbf{w}), \tag{2}$$

where $n(\mathbf{X}(t); \mathbf{w}) : \mathbb{R} \rightarrow \mathbb{R}^n$ is a neural network parameterized by a weight vector $\mathbf{w}$.

Lastly, we consider a hybrid model, integrating both first-principles and neural components of the form

$$\frac{d^k\mathbf{X}(t)}{dt^k} = f(\mathbf{X}(t); \mathbf{p}) + n(\mathbf{X}(t); \mathbf{w}). \tag{3}$$

Here, we limit ourselves to the above additive form, but other possibilities such as multiplicative or nested forms are of course possible [27]–[29]. Central to the proposed approach is that the parameter vectors $\mathbf{p}$ and $\mathbf{w}$ are estimated simultaneously. As a consequence, there is a tight coupling between both components allowing for synergies. This is in contrast to other approaches where a data-driven component is trained on the residual error of the solution of a first-principles model [30]. Incorporating domain knowledge into the model through a first-principles component restricts the solution space to a desired subset, reducing overfitting through a regularization effect. When (a piece of) domain knowledge is incorporated, only its parameters need to be estimated from data, instead of its entire structure, decreasing the data requirements.

As the first-principles and neural components are trained simultaneously, the estimated parameters of both components influence each other. As a result, the parameters of the first-principles model $\mathbf{p}$ will lose part of their physical interpretation and should only be interpreted within the context of the hybrid model.

## II. TRAINING DIFFERENTIAL EQUATIONS WITH A NEURAL COMPONENT

Different methods can be used to identify the parameters of the right-hand side of a differential equation, whether it includes a first-principles, a neural component, or both. First-principles differential equations rely on a small number of parameters, most of which have a physical meaning. The majority of these parameters are known or deduced through experiments and subsequently used in the model. Parameters that cannot be estimated separately are fitted by minimizing a loss function $L$ between the data and the solution (i.e., numerical integration) of the differential equation.

Including a neural component substantially raises the number of parameters that need to be estimated, as the entire structure of the dynamics needs to be inferred. Consequently, this increases the complexity of training, requiring techniques capable of efficiently estimating a large number of parameters. We briefly discuss several options below, for a visualization of these methods, we refer to the Fig. SI 1.

### A. BACKPROPAGATION THROUGH THE SOLVER

The minimization of the loss function $L$ can be guided by its gradient with respect to all parameters, i.e. $\mathbf{p}$ and $\mathbf{w}$. This approach is called backpropagation through the solver, as the gradient is dependent on the output of the numerical solver.

This gradient can be estimated through the finite differences method or automatic differentiation. The former is generally used in first-principles differential equations, where typically only a few parameters need to be estimated. However, it scales badly with problem size, requiring additional function evaluations for each dimension. This makes the finite differences method infeasible to fit the many parameters of neural differential equations.

Alternatively, modern libraries such as Tensorflow [31] and PyTorch [32] can apply automatic differentiation (either forward or backward) over arbitrary functions, yielding the necessary gradients. Implementing a solver in one of these libraries allows for backpropagation through the solver without the overhead and scalability issues of finite differences. However, the solver needs to be reimplemented into such a library, making it impossible to reuse existing solvers such as the widely used SUNDIALS solver suite [33]. Besides, all operations of the solver need to be traced back to perform automatic differentiation, making this method memory intensive and less scalable for long time series.

### *B. ADJOINT METHOD*

The adjoint method for sensitivity analysis [34] has been adapted to compute the desired gradient in dynamical models [25], [35]. An adjoint is defined as the gradient of the loss with respect to the state $\mathbf{a}(t) = \frac{\partial L}{\partial \mathbf{X}(t)}$. The adjoint dynamics are given by the ordinary differential equation

$$\frac{d\mathbf{a}(t)}{dt} = -\mathbf{a}(t)^{\intercal} \frac{\partial \left[ f(\mathbf{X}(t); \mathbf{p}) + n(\mathbf{X}(t); \mathbf{w}) \right]}{\partial \mathbf{X}}, \tag{4}$$

which is analogous to the chain rule. By augmenting the state vector with this adjoint, the solver can compute the desired gradient through a second backward call. This allows the computation of the desired gradients with $\mathcal{O}(1)$ memory, at the expense of an additional call to the solver [25]. Moreover, as this method treats the solver as a black box, there is no need to reimplement it in a specialized library, making it possible to use the adjoint method with existing solvers. In dynamical systems with a large number of parameters, the adjoint method has been found to be more efficient than automatic differentiation [36], [37].

### *C. GRADIENT MATCHING*

Both methods described above require an explicit solution of the equations to calculate the error with respect to the data. At each iteration of the optimization process, an expensive call to the solver is thus needed. Moreover, errors get integrated through time and magnified by the solver, giving relatively higher importance to the earlier datapoints. In contrast, gradient matching tries to minimize the difference between calculated derivatives and the right-hand side of the differential equations. In this way, it directly learns the underlying differential equations, instead of its solution. This circumvents the need for an explicit solution at each iteration but introduces the need to calculate derivatives from data through numerical differentiation, greatly amplifying the noise in the data. Special methods based on spline interpolation [38] or total variation regularization [39]–[41] can be used to perform differentiation on noisy and nonsmooth data, yielding adequate results that can be used for gradient matching. When calculated derivatives are available, gradient matching scales very well to large systems.

## III. EXPERIMENTS

We demonstrate the proposed approach on several nonlinear systems, ranging from noisy measurements of a chaotic elastic pendulum system, over an unknown reaction in glycolysis, to a reaction-diffusion system, extending the framework to nonlinear partial differential equations. In each example, we explore the ability to identify unmodeled dynamics from only noisy state measurements.

The neural component of the models was a multilayer perceptron with hyperbolic tangent activation functions, with output dimension equal to the dimension of the system. Weights were initialized with mean $\mu = 0$ and standard deviation $\sigma = 0.1$. Importantly, the initialized $\sigma$ determines the balance between first-principles and neural component, with larger values of $\sigma$ giving rise to a more dominant neural component. Bias units were initialized to 0. The parameters of the first-principles component $\mathbf{p}$ were initialized based on the calibrated first-principles model. In all experiments, the size and number of hidden layers was chosen to balance accuracy (i.e. training and test error) and complexity.

Training was performed in a batched manner. At each iteration, a batch of a certain number of states was randomly chosen from the dataset. These were used as initial conditions for the model, integrating the solution for a certain amount of time using a fourth-order Runge-Kutta fixed stepsize scheme. A fixed stepsize scheme is preferred here as it allows parallelization of the batches. Moreover, as neural components can be unstable during training, the stepsize can become inhibitingly small when using an adaptive stepsize solver, causing arithmetic underflows. Subsequently, the root mean square error (RMSE) is calculated between the solutions and the data, with the desired gradient calculated using either automatic differentiation or the adjoint method. An ADAM method with learning rate 0.01 was used to optimize RMSE for 3000 iterations, with the learning rate lowered to 0.001 after 2000 iterations. Additional details of the experimental setup are reported for each experiment below.

As described above, gradients can be calculated with different methods, each with their advantages and drawbacks. For the examples presented in this work, we used the adjoint method, as this method is less sensitive to noise than gradient matching and more flexible and memory-efficient than automatic differentiation. All code was implemented in Python 3.7 using the PyTorch [32] and TorchDiffEq library [25] and executed on an Nvidia Pascal P100 accelerator. The source code used in this work is

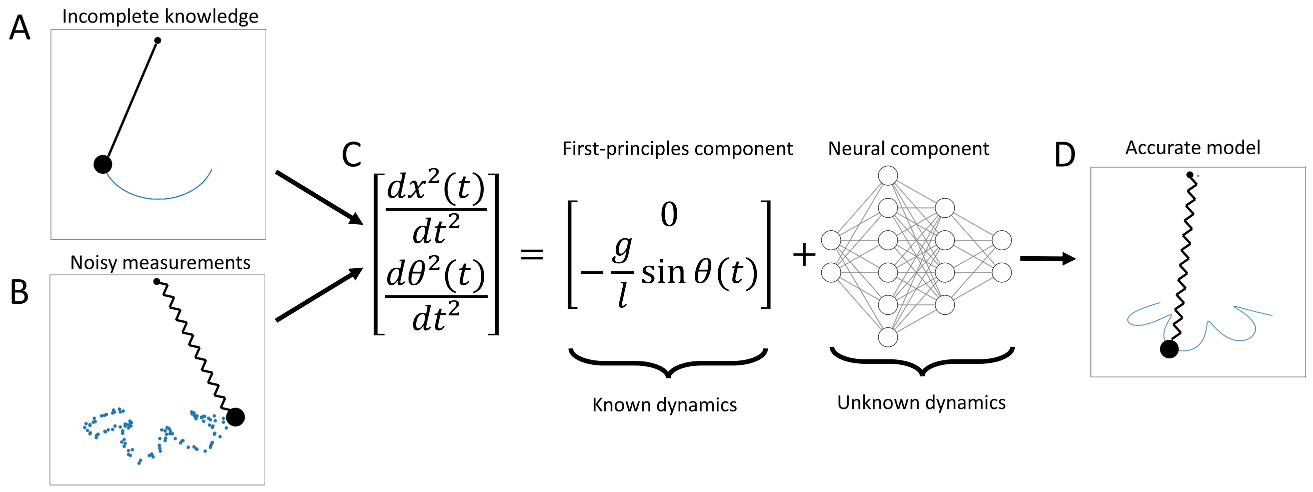

A
Incomplete knowledge
B
Noisy measurements
C
$\begin{bmatrix}\frac{dx^2(t)}{dt^2}\\ \frac{d\theta^2(t)}{dt^2}\end{bmatrix} = \begin{bmatrix}0\\ -\frac{g}{l}\sin\theta(t)\end{bmatrix} +$
First-principles component
Neural component
Known dynamics
Unknown dynamics
D
Accurate model

**FIGURE 1. Schematic of the proposed approach on an elastic pendulum system, where it is not known that the connecting thread is elastic. (A) Incomplete first-principles model of the system, based on the assumption that the system behaves as an ideal pendulum. (B) Noisy measurements of the system. The first-principles model does not accurately capture the dynamics underlying the data. (C) The proposed approach incorporates a first-principles component, encoding the known dynamics, and a neural component, capturing the unknown dynamics. (D) This results in a model that accurately captures the complete dynamics of the system.**

available at GitHub (https://github.com/WardQ/IncorporatingUnmodeledDynamics).

### A. ELASTIC PENDULUM

As a first example, we consider an elastic pendulum (Fig. 1). When an object is connected to a spring, the resulting motion exhibits dynamics of both a simple pendulum and an elastic system [42]. The system exhibits chaotic behavior and is sensitive to initial conditions. The physical knowledge of the elastic pendulum can be incorporated in a system of first-principles differential equations

$$\begin{aligned}\frac{d^2x(t)}{dt^2} &= (l_0 + x(t))\dot{\theta}(t)^2 - \frac{k}{m}x(t) + g\cos\theta(t),\\ \frac{d^2\theta(t)}{dt^2} &= -\frac{g}{l_0 + x(t)}\sin\theta(t) - \frac{2\dot{x}(t)}{l_0 + x(t)}\dot{\theta}(t),\end{aligned} \tag{5}$$

where $x(t)$ is the compression or extention of the spring at $t$, $\theta(t)$ is the angle of oscillation, $\dot{x}$ and $\dot{\theta}$ are first derivatives of respectively $x$ and $\theta$, $l_0$ the rest length, $k$ the spring constant, $m$ the mass of the pendulum, and $g$ the gravitational constant. A full derivation is given in SI Appendix 1. We consider a system with limited elasticity. Noisy measurements of an example trajectory are visualized in Fig. 1B.

For illustrative purposes, we consider a situation where the elastic dynamics are unknown and not incorporated into the first-principles model. This incomplete knowledge results in an ideal pendulum

$$\begin{aligned}\frac{d^2x(t)}{dt^2} &= 0,\\ \frac{d^2\theta(t)}{dt^2} &= -\frac{g}{l}\sin\theta(t).\end{aligned} \tag{6}$$

An example trajectory of this ideal pendulum is visualized in Fig. 1A. This situation, where a complex system is modeled by a simpler model that does not capture all underlying dynamics, is ubiquitous, either because these unmodeled dynamics are unknown or deliberately not included. Here, we apply a hybrid model (Eq. 3) incorporating the ideal pendulum (Eq. 6) as a first-principles component and a neural component that will learn the elastic dynamics from noisy data. More specifically, the neural component will learn the terms in Eq. (5) that are not present in Eq. (6).

In this case study, we sampled a trajectory of the elastic pendulum system with rest length $l_0 = 2.25$, spring constant $k = 200$, mass of the pendulum $m = 0.547$, and gravitational constant $g = 9.81$ for $t$ in [0, 10] at a sampling rate of 100 samples/s, with initial conditions $x_0 = -0.75$ and $\theta_0 = 1.25$ and added Gaussian noise $\sigma = 0.1$.

We subsequently fitted an (incomplete) first-principles, a neural, and a hybrid model to these data. Training was performed using batches of 256 states integrated over the next 16 timepoints. A neural component with two hidden layers with 40 and 20 neurons, respectively, was used. Training took 8 minutes on our hardware. During inference, the neural component adds a small overhead in comparison to the first-principles model.

The incomplete first-principles model cannot capture the elastic dynamics, as its structure only incorporates the dynamics of an ideal pendulum (Fig. SI 2). The learned dynamics reflects the period of the pendulum but not its elastic behavior. In contrast, both the neural and hybrid model capture the full dynamics of the elastic pendulum. The hybrid model estimates the length of the pendulum as $l = 0.9$. This approximates the term $l_0 + x(t)$ in Eq. (5), where $l_0 = 2.25$ and $x(t) \in [-1.5, -0.7]$. The deviation between the values

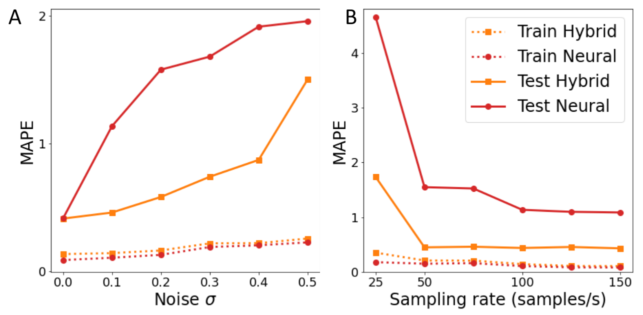


**FIGURE 2.** **MAPE on the training dataset, representing the extent to which the model has captured the dataset, and on the test dataset, representing the degree to which the model has captured the underlying dynamics and generalizes to unseen states. Results are given for both the hybrid model, incorporating a first-principles and neural component, and the neural model with only a neural component. (A) MAPE in function of the noise standard deviation $\sigma$ at 100 samples/s. Both models achieve a reasonably low training set error, even at high noise. However, the hybrid model consistently achieves a lower test error at higher noise levels. (B) MAPE in function of sampling rate at noise level $\sigma = 0.1$. Again, both models achieve a reasonably low training set error, even at low sampling rates. The hybrid model again consistently achieves a lower test set error. Both results emphasize the capabilities of the hybrid model to generalize to unseen states, better capturing the underlying dynamics.**

of $l$ and $l_0$ highlights the influence of the neural component on the first-principles component and the decreased interpretability of its parameters, as $l$ can only be interpreted alongside $x(t)$.

Besides capturing the dynamics of a given dataset, a model should capture the dynamics of the system and generalize to situations not seen before. To assess these capabilities, a neural and hybrid model were trained on several sampled datasets of the abovementioned trajectory with standard deviations $\sigma$ of Gaussian noise in $[0, 0.5]$ (Fig. SI 3) and sampling rate in $[25, 150]$ samples/s (Fig. SI 4). We subsequently tested these trained models for different initial conditions (called the test dataset), calculating the mean absolute percentage error (MAPE) between the model output and the system dynamics, thus assessing whether the model can extrapolate to situations not present in the training dataset. The MAPE is defined as

$$\mathrm{MAPE} = \frac{1}{n}\sum_{t=0}^{n}\left|\frac{\mathbf{X}(t) - \hat{\mathbf{X}}(t)}{\mathbf{X}(t)}\right|, \tag{7}$$

where $n$ is the number of observations, $\mathbf{X}_t$ and $\hat{\mathbf{X}}_t$ the data and predicted value at time $t$, respectively. Additionally, we report the $L_{\text{inf}}$ norm of the absolute percentage error on the test set, i.e., the maximum error in the sampled interval.

First, this procedure was executed for different standard deviations $\sigma$ of the Gaussian noise (Fig. 2A). Both models achieve a low MAPE on the training dataset, confirming the approximation capabilities of the neural component. The neural model achieved a slightly lower MAPE, as the first-principles component of the hybrid model has a regularizing effect, preventing overfitting to the training data. When evaluating on the test dataset, the MAPE is significantly lower for the hybrid model. As the hybrid model can rely on the first-principles component for (part of the) learned dynamics, it generalizes better to conditions not seen before. Logically, with increasing $\sigma$, the performance worsens for both models. The $L_{\text{inf}}$ norm of the error exhibits a similar, magnified pattern (Fig. S5A). As errors get integrated over time, the $L_{\text{inf}}$ norm increases. At $\sigma \geq 0.5$, the noise becomes bigger than the elastic dynamics, making identification impossible. Interestingly, when trained on noiseless data ($\sigma = 0$), both models perform equally well, as no overfitting to noise occurs.

Second, the sampling rate is varied (Fig. 2B and S5B). Again, the hybrid model achieves a lower MAPE and $L_{\text{inf}}$ on the test set in comparison to the purely neural model and generalizes better to conditions not present in the training data. When lowering the sampling rate to 25 samples/s, the elastic dynamics become unidentifiable.

Both results show that incorporating knowledge into a data-driven model prevents overfitting to the training data, lowers requirements on quantity and quality of training data, and results in a model that generalizes better to unseen states.

## B. GLYCOLYSIS

In the elastic pendulum example, the first-principles model did not capture all underlying dynamics. Nevertheless, this simple model did capture the general trends and was, depending on the accuracy requirements, not entirely unusable. However, for many systems, e.g., those relying on energy or mass balances, incomplete knowledge leads to a model that completely fails. Here, we consider a glycolysis model, a benchmark problem for system identification [9], [44], [45]. A simplified version of the chemical reaction network of this system is given in Fig. 3A. A full description can be found in SI Appendix 2.

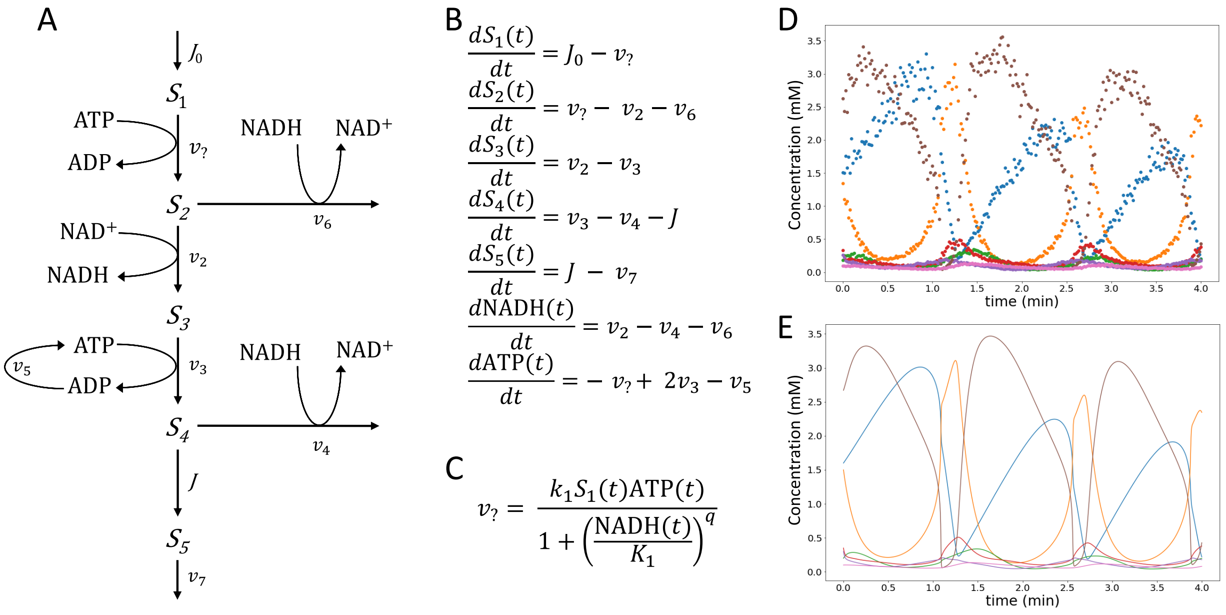
In this example, all reaction kinetics except $v_?$ are known. This missing piece of knowledge renders the complete model useless (Fig. SI 7), as all subsequent reactions are dependent on the reaction $S_1 + \text{ATP} \xrightarrow{v_?} S_2 + \text{ADP}$ and thus on $v_?$. Moreover, it is difficult to directly identify the unknown kinetics $v_?$ from the data, as the reaction involves ATP, which is also engaged in other reactions that are (indirectly) dependent on $v_?$.

Here, we consider a system where the unknown $v_?$ is governed by the nonlinear reaction rate

$$v_? = \frac{k_1 S_1(t)\text{ATP}(t)}{1 + \left(\frac{S_6(t)}{K_1}\right)^q}, \tag{8}$$

where $k_1$, $K_1$, and $q$ are reaction constants.

We sampled datapoints with scaled Gaussian noise $\sigma = 0.05$ from the glycolysis system for $t$ in $[0, 4]$ at 250 samples/s (Fig. 3B). Training was performed using batches of 256 states integrated over the next 16 time points and was completed within 17 minutes. A neural component with two hidden layers of 50 and 20 neurons, respectively, was used.



$$\frac{dS_1(t)}{dt} = J_0 - v_?$$
$$\frac{dS_2(t)}{dt} = v_? - v_2 - v_6$$
$$\frac{dS_3(t)}{dt} = v_2 - v_3$$
$$\frac{dS_4(t)}{dt} = v_3 - v_4 - J$$
$$\frac{dS_5(t)}{dt} = J - v_7$$
$$\frac{d\text{NADH}(t)}{dt} = v_2 - v_4 - v_6$$
$$\frac{d\text{ATP}(t)}{dt} = - v_? + 2v_3 - v_5$$

$$v_? = \frac{k_1 S_1(t)\text{ATP}(t)}{1 + \left(\frac{\text{NADH}(t)}{K_1}\right)^q}$$

**FIGURE 3. Model results on the glycolysis example. (A) Chemical reaction network of the system as defined in [43]. (B) In this example, the system is completely known, except for (C) the reaction kinetics $v_?$ of the reaction $S_1 + \text{ATP} \xrightarrow{v_?} S_2 + \text{ADP}$. (D) Noisy measurements of the glycolysis system. (E) The approach integrates both incomplete knowledge and data to accurately capture the complete dynamics of the system.**

The first-principles component encodes all available knowledge, whereas the neural component learns the dynamics of the unknown reaction (Eq. 8). This results in an integrated hybrid model that accurately captures all dynamics, achieving a MAPE of 0.092 and an $L_{\text{inf}}$ norm of 0.127 (Fig. 3C).

An additional experiment, similar to the previous one, was performed to assess the impact of the missing piece on the accuracy of the model. Here, the reaction kinetics $v_2$ is unknown, instead of $v_?$. The reaction kinetics $v_2$ is governed by

$$v_2 = k_2 S_2(t)\big(N - \text{NADH}(t)\big), \tag{9}$$

where $k_2$ and $N$ are reaction constants. In comparison to $v_?$, the reaction kinetics $v_2$ is relatively simple. Using the previously described experimental setup, the hybrid model achieves a MAPE of 0.058 and an $L_{\text{inf}}$ norm of 0.083, substantially lower than the previous experiment. The accuracy is thus higher when $v_2$ is missing than when $v_?$ is missing, showing a trade-off between the complexity of the missing dynamics and accuracy.

Up until now, we have initialized the parameters $\mathbf{p}$ of the first-principles component based on the available knowledge. To investigate the robustness to poor initialization, we randomly selected the components of $\mathbf{p}$ from the interval [0, 5]. The model was then trained on the dataset. This experiment was repeated 100 times. In 89 of these experiments, the first-principles component correctly captured the underlying dynamics, indicating a certain robustness. In the remaining 11 cases, the neural component suppressed the first-principles component, with the parameters $\mathbf{p}$ going to $\mathbf{0}$, thus eliminating this dynamics. This illustrates the possibility that the neural component learns structures that are explicitly defined in the first-principles component.

### C. BELOUSOV-ZHABOTINKSY REACTION-DIFFUSION

So far, we have considered ordinary differential equations, involving a single independent variable (usually time $t$). The proposed approach can be extended to partial differential equations, involving multiple independent variables, usually including a spatial component. As an example, we consider a system based on the Belousov-Zhabotinsky reaction, a classical example of non-equilibrium thermodynamics, resulting in the establishment of a nonlinear chemical oscillator [46], [47]. We consider a spatially dependent system governed by diffusion that exhibits intriguing spatial patterns and waves [48]. This reaction can be described by a system of partial differential equations involving the reagents $u$ and $v$

$$\begin{aligned} \epsilon \frac{\partial v(t)}{\partial t} &= D\nabla^2 u(t) + u(t)(1 - u(t)) - \frac{u(t) - q}{u(t) + q} v(t), \\ \frac{\partial v(t)}{\partial t} &= D\nabla^2 v(t) + u(t) - v(t), \end{aligned} \tag{10}$$

where $D$ is the diffusion coefficient, identical for $u$ and $v$, and $\epsilon$ and $q$ are reaction parameters.

Diffusion is a well-understood process that can be described by the diffusion equation (i.e., the first term of the right-hand side of (10), Fig. SI 10). Here we consider the situation where the reaction dynamics are unknown. Hence, the diffusion dynamics are explicitly modeled as a first-principles component, with the reaction terms captured by the neural component. Here, the reaction dynamics

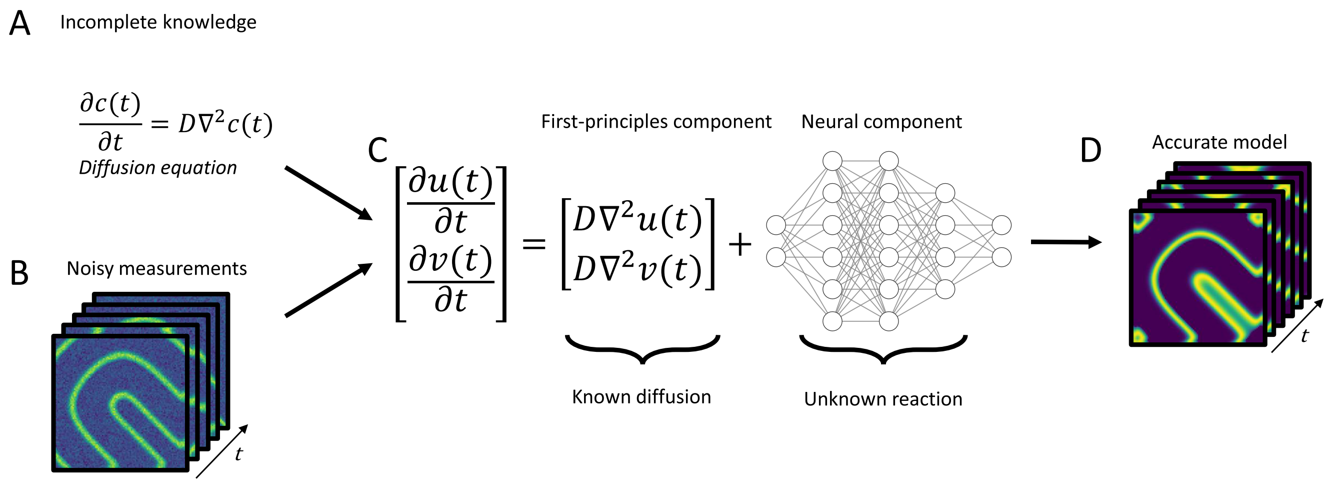

A Incomplete knowledge
$\frac{\partial c(t)}{\partial t} = D\nabla^2 c(t)$
Diffusion equation
B Noisy measurements
t
C
$\begin{bmatrix}\frac{\partial u(t)}{\partial t}\\ \frac{\partial v(t)}{\partial t}\end{bmatrix} = \begin{bmatrix}D\nabla^2 u(t)\\ D\nabla^2 v(t)\end{bmatrix} +$
First-principles component
Neural component
Known diffusion
Unknown reaction
D Accurate model
t

**FIGURE 4.** **Example of a spatial Belousov-Zhabotinksy reaction-diffusion system, where the reaction dynamics are unknown. The proposed approach incorporates (A) the available knowledge on diffusion dynamics and (B) spatial measurements of the system. (C) The diffusion is explicitly modeled by the first-principles component, whereas the reaction is approximated from the data. (D) This results in a model that accurately captures the complete dynamics of the system.**

are only time-dependent, but unknown spatial dependencies could be learned by convolutions [49].

As training data, we sampled datapoints with Gaussian noise $\sigma = 0.01$ from a Belousov-Zhabotinsky reaction-diffusion system with $D = 0.001$, $\epsilon = 0.2$, $q = 0.001$ for $t$ in [0, 25] at 100 samples/s. Spatial discretization was performed for $x \in [0, 5]$ and $y \in [0, 5]$ with no boundary conditions and $\Delta x = \Delta y = 0.05$.

We fitted a hybrid model with a neural component with three hidden layers with 50, 50, and 20 neurons, respectively, to this sampled data. Training was completed within 37 minutes. Again, the neural component correctly identifies the unknown reaction, resulting in a model that accurately describes all dynamics, achieving a MAPE of 0.083 and an $L_{\text{inf}}$ norm of 0.112 (Fig. 4D and S 11).

## IV. DISCUSSION

In summary, we have demonstrated a powerful approach to incorporate unmodeled dynamics into first-principles models. It builds on recent advances in data-driven system identification, more specifically incorporating an assumption that the data was sampled from a dynamical system [25], [26]. Our work extends these approaches by incorporating a first-principles component. First-principles models are usually based on domain knowledge that is simplified, incomplete, and subject to certain assumptions. Dynamics that are not modeled explicitly by the first-principles component are captured by the machine learning component, thereby filling in knowledge gaps. We have demonstrated this approach on several example systems exhibiting chaos, unknown chemical reactions, and spatial dependencies. As shown in the elastic pendulum example, this approach lowers the data requirements, as it is less sensitive to noise and large sampling intervals. As shown in the glycolysis example, this approach provides a representative model when incomplete knowledge inhibits the use of a first-principles model. Moreover, the approach is robust to poor initialization. The Belousov-Zhabotinsky example shows that this approach can be easily extended to incorporate spatial dynamics.

We have introduced several methods to train differential equations with a neural component. In each case, the hybrid model identified unmodeled dynamics from noisy data, resulting in a model that is representative of the whole system. Although we have limited ourselves to an additive combination of both components, the methodology can likely be ported to other combinations. For example, in a system where the ideal dynamics are known but impacted by an unknown resistance, the first-principles component can be multiplied by a neural component capturing this resistance [50]. Hence, domain knowledge of the system can be incorporated by changing the architecture. Moreover, this framework should not be limited to a neural component. Other types of data-driven components, such as radial basis functions or polynomials, could be preferred depending on the complexity of the missing dynamics and the use case.

The approach introduces a black box into the model, decreasing the interpretability and physical relevance of the parameters of the first-principles component. There is thus a trade-off between accuracy and interpretability. This is in contrast with other techniques such as symbolic [4], [5] or sparse regression [9], [10] that are aimed at distilling interpretable models, but suffer from scalability issues or are limited by the predefined library of candidate functions, respectively. Therefore, we foresee the use of this approach mainly in applications where predictive capabilities are detrimental, e.g., in predictive control. Additionally, the integrated model can easily adapt to changes in the dynamics in an online learning setting. Nevertheless, this approach cannot be applied for designing new systems, as no data is yet available. Last but not least, analysis of the neural component can potentially direct the modeller towards gaps in the first-principles model, catalyzing model improvements.

As with all data-driven techniques, one needs to be cautious about data quality. Statistics and machine learning practitioners have developed methods to counteract overfitting to noisy data (e.g., regularization) and to properly evaluate the model (e.g., cross-validation). These methods should be incorporated into good modeling practice.

This approach can be applied in numerous fields with incomplete knowledge but plenty of data on the system, including climate science, chemical installations, epidemiology, and financial markets. Neither first-principles nor data-driven modeling is the solution for all challenges in science and engineering. Integrating both combines their advantages while mitigating their disadvantages. In conclusion, the identification and incorporation of unmodeled dynamics is an important step towards modeling any dynamical system.

## ACKNOWLEDGMENT

The authors would like to thank insightful discussions with Bram De Jaegher.

## REFERENCES

[1] L. Perko, *Differential Equations and Dynamical Systems*, vol. 7. Springer, 2013.

[2] R. Woods and K. Lawrence, *Modeling and Simulation of Dynamic Systems* (Prentice-Hall Series in Geographic). Upper Saddle River, NJ, USA: Prentice-Hall, 1997. [Online]. Available: https://books.google.be/books?id=TIRRAAAAMAAJ

[3] M. Braun and M. Golubitsky, *Differential Equations and Their Applications*, vol. 1. Springer, 1983.

[4] J. Bongard and H. Lipson, "Automated reverse engineering of nonlinear dynamical systems," *Proc. Nat. Acad. Sci. USA*, vol. 104, no. 24, pp. 9943–9948, Jun. 2007.

[5] M. Schmidt and H. Lipson, "Distilling free-form natural laws from experimental data," *Science*, vol. 324, no. 5923, pp. 81–85, Apr. 2009.

[6] M. Quade, M. Abel, K. Shafi, R. K. Niven, and B. R. Noack, "Prediction of dynamical systems by symbolic regression," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 94, no. 1, Jul. 2016, Art. no. 012214.

[7] J. R. Koza and J. R. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection*, vol. 1. Cambridge, MA, USA: MIT Press, 1992.

[8] H. Cao, L. Kang, Y. Chen, and J. Yu, "Evolutionary modeling of systems of ordinary differential equations with genetic programming," *Genetic Program. Evolvable Mach.*, vol. 1, no. 4, pp. 309–337, 2000.

[9] S. L. Brunton, J. L. Proctor, and J. N. Kutz, "Discovering governing equations from data by sparse identification of nonlinear dynamical systems," *Proc. Nat. Acad. Sci. USA*, vol. 113, no. 15, pp. 3932–3937, Apr. 2016.

[10] H. Schaeffer, G. Tran, and R. Ward, "Extracting sparse high-dimensional dynamics from limited data," *SIAM J. Appl. Math.*, vol. 78, no. 6, pp. 3279–3295, Jan. 2018.

[11] S. Chen, S. A. Billings, C. F. N. Cowan, and P. M. Grant, "Non-linear systems identification using radial basis functions," *Int. J. Syst. Sci.*, vol. 21, no. 12, pp. 2513–2539, Dec. 1990.

[12] J. Kocijan, A. Girard, B. Banko, and R. Murray-Smith, "Dynamic systems identification with Gaussian processes," *Math. Comput. Model. Dyn. Syst.*, vol. 11, no. 4, pp. 411–424, Dec. 2005.

[13] M. Raissi, P. Perdikaris, and G. E. Karniadakis, "Machine learning of linear differential equations using Gaussian processes," *J. Comput. Phys.*, vol. 348, pp. 683–693, Nov. 2017.

[14] H. Ye, R. J. Beamish, S. M. Glaser, S. C. H. Grant, C.-H. Hsieh, L. J. Richards, J. T. Schnute, and G. Sugihara, "Equation-free mechanistic ecosystem forecasting using empirical dynamic modeling," *Proc. Nat. Acad. Sci. USA*, vol. 112, no. 13, pp. E1569–E1576, Mar. 2015.

[15] S. A. Billings, *Nonlinear System Identification: NARMAX Methods in the Time, Frequency, and Spatio-Temporal Domains*. Hoboken, NJ, USA: Wiley, 2013.

[16] R. González-García, R. Rico-Martínez, and I. G. Kevrekidis, "Identification of distributed parameter systems: A neural net based approach," *Comput. Chem. Eng.*, vol. 22, pp. S965–S968, Mar. 1998.

[17] S. Masri, A. Chassiakos, and T. Caughey, "Structure-unknown non-linear dynamic systems: Identification through neural networks," *Smart Mater. Struct.*, vol. 1, no. 1, p. 45, 1992.

[18] M. Raissi, P. Perdikaris, and G. Em Karniadakis, "Multistep neural networks for data-driven discovery of nonlinear dynamical systems," 2018, *arXiv:1801.01236*. [Online]. Available: http://arxiv.org/abs/1801.01236

[19] M. Ayoubi, "Nonlinear dynamic systems identification with dynamic neural networks for fault diagnosis in technical processes," in *Proc. IEEE Int. Conf. Syst., Man Cybern.*, vol. 3, Oct. 1994, pp. 2120–2125.

[20] G. Cybenko, "Approximation by superpositions of a sigmoidal function," *Math. Control, Signals, Syst.*, vol. 2, no. 4, pp. 303–314, Dec. 1989.

[21] T. Chen and H. Chen, "Universal approximation to nonlinear operators by neural networks with arbitrary activation functions and its application to dynamical systems," *IEEE Trans. Neural Netw.*, vol. 6, no. 4, pp. 911–917, Jul. 1995.

[22] K.-I. Funahashi and Y. Nakamura, "Approximation of dynamical systems by continuous time recurrent neural networks," *Neural Netw.*, vol. 6, no. 6, pp. 801–806, Jan. 1993.

[23] X.-D. Li, J. K. L. Ho, and T. W. S. Chow, "Approximation of dynamical time-variant systems by continuous-time recurrent neural networks," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 52, no. 10, pp. 656–660, Oct. 2005.

[24] O. De Jesus and M. T. Hagan, "Backpropagation algorithms for a broad class of dynamic networks," *IEEE Trans. Neural Netw.*, vol. 18, no. 1, pp. 14–27, Jan. 2007.

[25] T. Q. Chen, Y. Rubanova, J. Bettencourt, and D. K. Duvenaud, "Neural ordinary differential equations," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 6571–6583.

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

[27] D. C. Psichogios and L. H. Ungar, "A hybrid neural network-first principles approach to process modeling," *AIChE J.*, vol. 38, no. 10, pp. 1499–1511, Oct. 1992.

[28] H.-T. Su, N. Bhat, P. Minderman, and T. McAvoy, "Integrating neural networks with first principles models for dynamic modeling," in *Dynamics and Control of Chemical Reactors, Distillation Columns and Batch Processes*. Amsterdam, The Netherlands: Elsevier, 1993, pp. 327–332.

[29] R. Oliveira, "Combining first principles modelling and artificial neural networks: A general framework," *Comput. Chem. Eng.*, vol. 28, no. 5, pp. 755–766, May 2004.

[30] R. J. Patton, "Fault diagnosis in nonlinear dynamic systems via neural networks," in *Proc. Int. Conf. Control*, 1994, p. 1346.

[31] M. Abadi *et al.* (2015). *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. [Online]. Available: https://www.tensorflow.org/

[32] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, and A. Desmaison, "Pytorch: An imperative style, high-performance deep learning library," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 8026–8037.

[33] A. C. Hindmarsh, P. N. Brown, K. E. Grant, S. L. Lee, R. Serban, D. E. Shumaker, and C. S. Woodward, "Sundials: Suite of nonlinear and differential/algebraic equation solvers," *ACM Trans. Math. Softw.*, vol. 31, no. 3, pp. 363–396, 2005.

[34] L. S. Pontryagin, E. Mishchenko, V. Boltyanskii, and R. Gamkrelidze, "The mathematical theory of optimal processes," Tech. Rep., 1962.

[35] G. Allaire, "A review of adjoint methods for sensitivity analysis, uncertainty quantification and optimization in numerical codes," *Ingenieurs de l'Automobile*, vol. 836, pp. 33–36, Jul. 2015.

[36] B. Sengupta, K. J. Friston, and W. D. Penny, "Efficient gradient computation for dynamical models," *NeuroImage*, vol. 98, pp. 521–527, Sep. 2014.

[37] C. Rackauckas, Y. Ma, V. Dixit, X. Guo, M. Innes, J. Revels, J. Nyberg, and V. Ivaturi, "A comparison of automatic differentiation and continuous sensitivity analysis for derivatives of differential equation solutions," 2018, *arXiv:1812.01892*. [Online]. Available: http://arxiv.org/abs/1812.01892

[38] P. Dierckx, "An algorithm for smoothing, differentiation and integration of experimental data using spline functions," *J. Comput. Appl. Math.*, vol. 1, no. 3, pp. 165–184, Sep. 1975.
[39] L. I. Rudin, S. Osher, and E. Fatemi, "Nonlinear total variation based noise removal algorithms," *Phys. D, Nonlinear Phenomena*, vol. 60, nos. 1–4, pp. 259–268, 1992.
[40] R. Chartrand, "Numerical differentiation of noisy, nonsmooth data," *ISRN Appl. Math.*, vol. 2011, pp. 1–11, May 2011.
[41] R. Chartrand, "Numerical differentiation of noisy, nonsmooth, multidimensional data," in *Proc. IEEE Global Conf. Signal Inf. Process. (GlobalSIP)*, 2017, pp. 244–248.
[42] W. K. Lee and H. D. Park, "Chaotic dynamics of a harmonically excited spring-pendulum system with internal resonance," *Nonlinear Dyn.*, vol. 14, no. 3, pp. 211–229, 1997.
[43] P. Ruoff, M. K. Christensen, J. Wolf, and R. Heinrich, "Temperature dependency and temperature compensation in a model of yeast glycolytic oscillations," *Biophysical Chem.*, vol. 106, no. 2, pp. 179–192, 2003.
[44] M. D. Schmidt, R. R. Vallabhajosyula, J. W. Jenkins, J. E. Hood, A. S. Soni, J. P. Wikswo, and H. Lipson, "Automated refinement and inference of analytical models for metabolic networks," *Phys. Biol.*, vol. 8, no. 5, 2011, Art. no. 055011.
[45] B. C. Daniels and I. Nemenman, "Efficient inference of parsimonious phenomenological models of cellular dynamics using S-systems and alternating regression," *PLoS ONE*, vol. 10, no. 3, Mar. 2015, Art. no. e0119821.
[46] R. J. Field, *Oscillations and Traveling Waves in Chemical Systems*. Hoboken, NJ, USA: Wiley, 1985.
[47] A. B. Rovinskii and A. M. Zhabotinskii, "Mechanism and mathematical model of the oscillating bromate-ferroin-bromomalonic acid reaction," *J. Phys. Chem.*, vol. 88, no. 25, pp. 6081–6084, Dec. 1984.
[48] V. K. Vanag, A. M. Zhabotinsky, and I. R. Epstein, "Oscillatory clusters in the periodically illuminated, spatially extended Belousov-Zhabotinsky reaction," *Phys. Rev. Lett.*, vol. 86, no. 3, p. 552, 2001.
[49] Z. Long, Y. Lu, X. Ma, and B. Dong, "PDE-Net: Learning PDEs from data," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 3208–3216.
[50] B. De Jaegher, E. Larumbe, W. De Schepper, A. Verliefde, and I. Nopens, "Colloidal fouling in electrodialysis: A neural differential equations model," *Separat. Purification Technol.*, vol. 249, Oct. 2020, Art. no. 116939.

**WARD QUAGHEBEUR** received the B.S. and M.S. degrees in bioscience engineering from Ghent University, Belgium, in 2015 and 2017, respectively. He is currently pursuing the Ph.D. in mathematical modelling with Ghent University. He was a Researcher with EUTEC, Hochschule Emden/Leer and also with the Technical University of Oruro, Bolivia. His research interests include dynamical systems modelling, machine learning, and bioprocesses.

**INGMAR NOPENS** received the M.S. and Ph.D. degrees in bioscience engineering from Ghent University, Belgium. He is currently a Full Professor with the BIOMATH Research Group, Ghent University, where he is also working on model-based analysis and optimisation of (bio)processes. His current research interests include mathematical modeling using different frameworks like biokinetics, computational fluid dynamics and population balance models, and combinations thereof. This is applied to a variety of processes in the fields of resource recovery and pharmaceutical engineering.

**BERNARD DE BAETS** received the M.Sc. degree in mathematics, the master's degree in knowledge technology, and the Ph.D. degree in mathematics from Ghent University, Belgium, in 1988, 1991, and 1995, respectively. He is currently a Senior Full Professor with Ghent University, where he is also leading the research unit Knowledge-Based Systems (KERMIT) and the Department of Data Analysis and Mathematical Modelling. He has acted as a supervisor of more than 75 Ph.D. students and has published over 550 peer-reviewed journal articles. He has delivered more than 300 (invited) conference lectures.

He is a Fellow of the International Fuzzy Systems Association. He was a recipient of the EUSFLAT Scientific Excellence Award, an Honorary Professor of Budapest Tech, a Doctor Honoris Causa with the University of Turku, a Professor Invitado of the Universidad Central "Marta Abreu" de Las Villas in Cuba and also a Professor Extraordinarius with the University of South Africa. He is a Government of Canada Award holder and has been nominated for the Ghent University Prometheus Award for Research. At present, he is Co-Editor-in-Chief of *Fuzzy Sets and Systems* and a member of the Editorial Board of several other journals.