Journal of Computational Physics 468 (2022) 111466

Contents lists available at ScienceDirect

Journal of Computational Physics

www.elsevier.com/locate/jcp

Stochastic physics-informed neural ordinary differential equations

Jared O'Leary [a], Joel A. Paulson [b], Ali Mesbah [a,*]

[a] Department of Chemical and Biomolecular Engineering, University of California, Berkeley, 94720, CA, USA
[b] Department of Chemical and Biomolecular Engineering, The Ohio State University, Columbus, 43210, OH, USA

ARTICLE INFO

Article history:
Received 28 August 2021
Received in revised form 14 May 2022
Accepted 6 July 2022
Available online 18 July 2022

Keywords:
Stochastic differential equations
Neural ordinary differential equations
Physics-informed neural networks
Moment-matching
Hidden physics
Uncertainty propagation

ABSTRACT

Stochastic differential equations (SDEs) are used to describe a wide variety of complex stochastic dynamical systems. Learning the hidden physics within SDEs is crucial for unraveling fundamental understanding of these systems' stochastic and nonlinear behavior. We propose a flexible and scalable framework for training artificial neural networks to learn constitutive equations that represent hidden physics within SDEs. The proposed stochastic physics-informed neural ordinary differential equation framework (SPINODE) propagates stochasticity through the known structure of the SDE (i.e., the known physics) to yield a set of deterministic ODEs that describe the time evolution of statistical moments of the stochastic states. SPINODE then uses ODE solvers to predict moment trajectories. SPINODE learns neural network representations of the hidden physics by matching the predicted moments to those estimated from data. Recent advances in automatic differentiation and mini-batch gradient descent with adjoint sensitivity are leveraged to establish the unknown parameters of the neural networks. We demonstrate SPINODE on three benchmark *in-silico* case studies and analyze the framework's numerical robustness and stability. SPINODE provides a promising new direction for systematically unraveling the hidden physics of multivariate stochastic dynamical systems with multiplicative noise.

1. Introduction

Stochastic dynamical systems are ubiquitous in a wide range of science and engineering problems, such as dynamical systems governed by Brownian motion or those that experience random perturbations from their surrounding environment [1–5]. Stochastic differential equations (SDEs) are used to describe the complex behavior of a wide variety of stochastic dynamical systems, including those involving electrical and cell signal processing [6–8], colloidal/molecular self-assembly [9,10], nucleation processes [11,12], and predator-prey dynamics [13,14]. An important challenge in constructing and studying SDEs is that they often contain physics that are either unknown or cannot be directly measured (e.g., free energy and diffusion landscapes [15,16], transmission functions in models of disease spread [17,18], etc.). Creating a systematic framework to learn the hidden physics within SDEs is thus crucial for unraveling fundamental understanding of stochastic dynamical systems.

* Corresponding author.
E-mail addresses: jared.oleary@berkeley.edu (J. O'Leary), paulson.82@osu.edu (J.A. Paulson), mesbah@berkeley.edu (A. Mesbah).

https://doi.org/10.1016/j.jcp.2022.111466
0021-9991/

A fairly general representation of SDEs is given by:

$$dx = f(x, g(x))dt + h(x, g(x))dw, \tag{1}$$

where x is the system state that is generally vector-valued, t is the time, and w is generally a multivariable Gaussian white noise process. The "modeled" or "known" physics is comprised of $f(\cdot)$, $h(\cdot)$, and the structure of the SDE (i.e., the additive relationship between $f(\cdot)$ and $h(\cdot)$ and the multiplicative relationship between $h(\cdot)$ and w). In this work, we consider $g(x)$ to be the "unmodeled" or "unknown" hidden physics. We thus seek to investigate strategies to create a flexible and scalable framework for systematically learning the hidden physics $g(x)$ within SDEs of form Eq. (1) from stochastic trajectory data.

The most commonly reported methods for learning $g(x)$ from stochastic trajectory data involve evaluating the time limits of the first and second conditional moments [19–30]:

$$f(x, g(x)) = \lim_{\tau \to 0} \frac{1}{\tau} \Big\langle (\xi(t+\tau) - \xi(t)) | \xi(t) = x \Big\rangle, \tag{2a}$$

$$h(x, g(x)) = \lim_{\tau \to 0} \frac{1}{2\tau} \Big\langle (\xi(t+\tau) - \xi(t))^2 | \xi(t) = x \Big\rangle, \tag{2b}$$

where ξ denotes a realization of the stochastic process with a δ-function distribution at the starting point t, $\xi(t) = x$, τ is the sampling time, and the angular brackets denote ensemble averaging. In practice, $\tau \to 0$ must be extrapolated or τ must be chosen to be sufficiently small to represent the limit. As the lower bound of τ is often determined by experimental limitations, the primary challenge facing works [22–30] is how to determine a robust way to extrapolate $\tau \to 0$. Common approaches to address this challenge involve adding correction terms to Eq. (2b) [22], using autocorrelation functions to simplify Eq. (2b) [31], employing kernel-based regressions over τ [23], and iteratively updating the limit evaluations based on computed probability distributions [25]. However, such methods generally rely on inflexible, data-intensive, and system-specific sampling techniques and/or have been shown to be non-viable when short time linear regions do not exist in the trajectory data [15,16,23].

Alternative approaches for learning $g(x)$ leverage Bayesian inference to estimate transition rates along adjacent intervals of x, e.g., [10,15,16,32–36]. The hidden physics $g(x)$ can then be recovered by exploiting relationships derived from the Fokker-Planck equation [37]. Although Bayesian inference approaches have been shown to be less sensitive to the sampling time than those that depend on extrapolating $\tau \to 0$ [15,16], these approaches either (i) learn $g(x)$ at discrete values of x and then fit analytic functions to these discrete values [10,15,16,32–34], or (ii) represent the unknown $g(x)$ using basis functions and learn the coefficients of those basis functions [35,36]. The former approach can become intractable when the dimension of x is large, or when $g(x)$ is highly nonlinear and thus requires x to be finely discretized. The latter approach can be highly sensitive to the choice of basis functions and can exhibit other numerical issues. As such, this latter approach often requires *a priori* knowledge about the stochastic system to inform the choice of basis functions [35].

To address the shortcomings described above, we propose a new framework for learning the hidden physics $g(x)$ in Eq. (1), which we refer to as stochastic physics-informed neural ordinary differential equations (SPINODE). SPINODE approximates $g(x)$ as an artificial neural network, where the weights and biases within the neural network represent the SDE hidden physics. Artificial neural networks provide a scalable and flexible way of approximating the potentially highly nonlinear relationship between $g(x)$ and continuous values of x without the need for *a priori* assumptions about the form of that relationship [38–40]. SPINODE then combines the notions of neural ordinary differential equations (neural ODEs) [41,42] and physics-informed neural networks (PINN) [43–47] to learn the weights and biases within the neural network that approximates $g(x)$ from state trajectory data. If we had access to the true state distribution at particular time points (which is generally non-Gaussian due to the nonlinear terms appearing in Eq. (1)), we could attempt to identify the neural network parameters that minimize a distributional loss function (e.g., the sum of the Kullback–Leibler divergence between the true and predicted distribution). However, not only would this loss function be more complicated to evaluate, we often do not have direct access to exact state distributions since these must be estimated from a finite set of state trajectories collected from simulations or experiments. Therefore, we opt for a more tractable *moment-matching* framework [48–50] in this work, which is an established method in statistics for simplifying the distribution matching problem. There are two key advantages to the moment-matching approach in the context of partially known SDEs:

- We only require moments of the state to be measured at discrete time points (with potentially varying sample times) from some known initial state distribution, which are easier to estimate than the full probability distribution or conditional moments.
- The predicted moments of the state based on Eq. (1) can be estimated using established uncertainty propagation techniques. As long as we can differentiate through the chosen uncertainty propagation method, we can use concepts from the neural ODE framework to compute derivatives needed for efficient training while preserving important features from the underlying SDE.

Although SPINODE can be adapted to handle a variety of different uncertainty propagation methods, we mostly focus on the unscented transform (UT) method [51–53] due to its ability to gracefully tradeoff between accuracy and computational efficiency. The UT method, when applied to Eq. (1), yields simple analytic expressions for the mean and covariance of

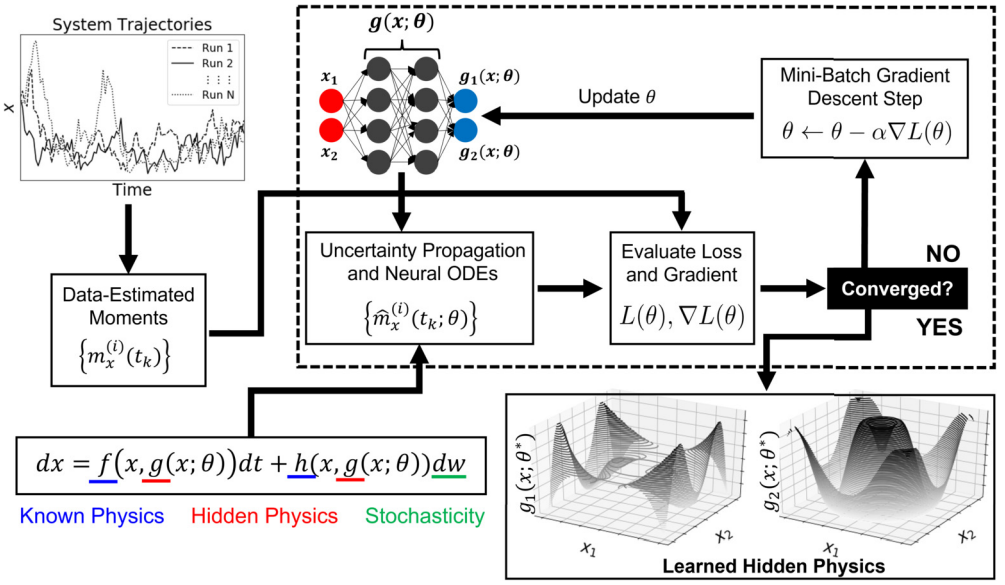

Fig. 1. Stochastic physics-informed neural ordinary differential equations framework (SPINODE). The key steps include (i) estimating the time evolution of statistical moments $m_x^{(i)}(t_k)$ from repeated stochastic dynamical system trajectories, (ii) approximating the hidden physics as a neural network (e.g., $g(x;\theta) = [g_1(x;\theta), g_2(x;\theta)]^\top$, where the unknown weights and biases are θ), (iii) using uncertainty propagation to propagate stochasticity through the known structure of the stochastic differential equation, (iv) using ODE solvers within the neural ODE framework to predict the time evolution of statistical moments $\hat{m}_x^{(i)}(t_k)$, and (v) using moment-matching and mini-batch gradient descent with adjoint sensitivity to learn "optimal" weights and biases θ^*. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

the states in terms of the solution to a relatively small set of ODEs. By defining and evaluating the model in terms of ODE solvers, we immediately gain the well-known benefits of such solvers including: (i) memory efficiency, (ii) adaptive computation with error control, and (iii) prediction at arbitrary sets of non-uniform time points [41]. All of these benefits are important when developing an efficient training algorithm for the neural network representation of $g(x)$ for which the loss gradient with respect to the neural network parameters can be computed using adjoint sensitivity methods [41,42,54].

To highlight the differences between SPINODE and previous methods, let us turn back to Eq. (2), which essentially computes the time derivative of the mean and covariance of x at some time t by some limit approximation. Previous works [19–30] have proposed many different strategies for interpolating measured state data from a finite set of discrete time points to estimate this limit; however, these strategies are largely system-specific. SPINODE, on the other hand, uses advanced uncertainty propagation and ODE solvers to directly predict state moment data at any set of time points. Since these underlying methods have been developed to apply to a diverse set of systems, including those that involve high-dimensional, nonlinear, and stiff dynamics, SPINODE can be flexibly applied to systems arising from all different types of applications, which we demonstrate by applying SPINODE to a variety of systems in this work. Furthermore, we note that although SPINODE is primarily described in the context of the first two moments in this paper for simplicity, it can naturally incorporate any number of moments (e.g., skew and kurtosis) when learning $g(x)$. This suggests that SPINODE has the potential to better handle highly non-Gaussian state distributions, which may arise when $f(\cdot)$, $h(\cdot)$, or $g(\cdot)$ are highly nonlinear.

We demonstrate the efficacy, flexibility, and scalability of SPINODE on three benchmark *in-silico* case studies. The dynamics of each system are described by SDEs of form Eq. (1) that contain nonlinear and state-dependent hidden physics terms. The first case study is a two-state model for directed colloidal self-assembly with an exogenous input [55], the second is a four-state competitive Lotka-Volterra model with a coexistence equilibrium [56], and the third is a six-state susceptible-infectious-recovered (SIR) epidemic model for disease spread [57]. We show that SPINODE is able to efficiently learn the hidden physics within these SDEs with high accuracy. We analyze the numerical robustness and stability of SPINODE and provide suggestions for future research. Furthermore, we have released a fully open-source version of SPINODE on GitHub with end-to-end examples [58], so that interested readers can easily reproduce and extend the results described in this work.

2. Methods

A schematic overview of the proposed SPINODE framework is shown in Fig. 1. Repeated stochastic dynamical system trajectories are recorded to estimate the time evolution of statistical moments of the stochastic state, $m_x^{(i)}(t_k)$ for all $i = 1, \ldots, N_m$ where N_m denotes the total number of moments considered (left). The hidden physics $g(x;\theta)$ are represented by a (deep) neural network that is parameterized by unknown weights and biases denoted by θ (center). Established uncertainty

propagation methods are used to propagate stochasticity through Eq. (1) and ODE solvers within the neural ODE framework are used to predict the time evolution of the moments for fixed neural network parameters, $\hat{m}_x^{(i)}(t_k;\theta)$ (left to center). A loss function is constructed using the predicted and data-estimated moments (center). Mini-batch gradient descent with adjoint sensitivity is used to update the parameters θ by minimizing the loss function (right). The hidden physics, $g(x;\theta)$, are considered "learned" once the mini-batch gradient descent algorithm converges. The subsequent subsections describe in more detail how data is collected and how SPINODE uses uncertainty propagation, neural ODEs, moment-matching, and mini-batch gradient descent to learn the weights and biases within the neural networks that approximate the unknown hidden physics within SDEs.

2.1. Data collection

Data collection is accomplished by repeating stochastic dynamical system trajectories starting from identical initial conditions. Here, N trajectories start from some initial condition x_0. During each trajectory, state values x are recorded at time points t_k for $K+1$ total time steps. The N recorded values of each $x(t_k)$ are used to estimate moments $m_x^{(i)}(t_k)$. For simplicity, we primarily focus on the first two moments, the state mean and covariance, which are calculated as follows:

$$\begin{aligned}\mu_x(t_k) &= \frac{1}{N}\sum_{n=1}^{N} x_n(t_k),\\ \Sigma_x(t_k) &= \frac{1}{N}\sum_{n=1}^{N}(x_n(t_k)-\mu_x(t_k))(x_n(t_k)-\mu_x(t_k))^\top,\end{aligned} \tag{3}$$

where $n \in \{1,\dots,N\}$ denotes the trajectory index.

Repeated stochastic trajectories from only one initial condition may not explore a large percentage of the state space. To compensate for this, the stochastic trajectories can be collected from multiple unique initial conditions. In this work, we choose initial conditions by performing a grid search within a specified range of state values of interest. We note, however, that more efficient sampling techniques, e.g., [59–63] can also be used, which will be explored in future work.

Although this work estimates moments of the stochastic states from repeated stochastic trajectories from identical initial conditions, we recognize that this strategy is not applicable for systems in which one does not have control over initial conditions, number of replica runs, or consistent measurement times. In such cases, probability distributions of state trajectories can be learned using methods that may not necessarily require such fine control over the observed trajectory data. Potentially suitable distribution estimation methods include variational autoencoders [64–66], generative adversarial networks [45,67–69], and/or energy-based models [70–72]. SPINODE is able to accommodate any data collection method from which the shape of the probability distribution (and thus the moments) can be estimated at discrete time points from observed trajectory data.

2.2. Moment prediction

As motivated in the introduction, an important advantage of the moment-matching framework is that we can rely on efficient uncertainty propagation methods that do not require access to the full distribution of the states. Unscented transform (UT) [51,52,73–75] is one such example of an efficient uncertainty propagation method that estimates moments from a set of well-placed samples (known as *sigma points*) that can be efficiently evaluated using a neural ODE solver.

Before applying UT to SDEs, let us first summarize the UT method for estimating the moments of a random variable $y = F(z)$ that is some static nonlinear transformation $F:\mathbb{R}^n \to \mathbb{R}^m$ of a random input $z \in \mathbb{R}^n$. We assume knowledge of the mean $m \in \mathbb{R}^n$ and covariance $P \in \mathbb{R}^{n\times n}$ of z. Given this information, UT involves the following 3 steps:

1. Form the set of $2n+1$ sigma points from the columns of the matrix $A = \sqrt{(n+\lambda P)}$, which denotes the Cholesky decomposition, as follows

$$\begin{aligned} z^{(0)} &= m, \\ z^{(i)} &= m + [A]_i, && i = 1,\dots,n, \\ z^{(i)} &= m - [A]_{n-i}, && i = n+1,\dots,2n, \end{aligned} \tag{4}$$

where $[A]_i$ denotes the i^{th} column of the matrix A. Then, compute the associated weights of each of these sigma points

$$\begin{aligned} W_0^{(m)} &= \frac{\lambda}{n+\lambda}, \\ W_0^{(c)} &= \frac{\lambda}{(n+\lambda)-(1-\alpha^2+\beta)}, \end{aligned} \tag{5}$$

$$W_i^{(m)} = \frac{1}{2(n+\lambda)}, \qquad i = 1, \ldots, 2n,$$
$$W_i^{(c)} = \frac{1}{2(n+\lambda)}, \qquad i = 1, \ldots, 2n,$$

where λ is a scaling factor defined by

$$\lambda = \alpha^2(n+\kappa) - n, \tag{6}$$

and α, β, and κ are positive constants. Typically, one should set α to be small (e.g., 10^{-3}), $\beta = 2$, and $\kappa = 0$ based on observations from [52].

2. Transform each of the sigma points as follows

$$y^{(i)} = F(z^{(i)}), \quad i = 0, \ldots, 2n. \tag{7}$$

3. Compute estimates for the mean and covariance of y

$$\hat{\mu}_y = \sum_{i=0}^{2n} W_i^{(m)} y^{(i)}, \tag{8}$$
$$\hat{\Sigma}_y = \sum_{i=0}^{2n} W_i^{(c)} \left(y^{(i)} - \hat{\mu}_y\right)\left(y^{(i)} - \hat{\mu}_y\right)^\top,$$

As shown in X, we can compactly represent UT in matrix form as follows

$$Z = \begin{bmatrix} m & \cdots & m \end{bmatrix} + \sqrt{\alpha^2(n+\kappa)}\begin{bmatrix} 0 & \sqrt{P} & -\sqrt{P} \end{bmatrix}, \tag{9}$$
$$Y = F(Z),$$
$$\hat{\mu}_y = Y w_m,$$
$$\hat{\Sigma}_y = Y W Y^\top,$$

where Z denotes the matrix of sigma points and $w_m \in \mathbb{R}^{2n+1}$ and $W \in \mathbb{R}^{2n+1\times 2n+1}$ are a vector and matrix defined in terms of the mean and covariance weight factors, respectively

$$w_m = [W_0^{(m)}, \ldots, W_{2n}^{(m)}]^\top, \tag{10}$$
$$W = \left(I - \begin{bmatrix} w_m & \cdots & w_m \end{bmatrix}\right) \mathrm{diag}(W_0^{(c)}, \ldots, W_{2n}^{(c)}) \left(I - \begin{bmatrix} w_m & \cdots & w_m \end{bmatrix}\right)^\top$$

and I denotes the identity matrix of appropriate size. This representation will be helpful when applying UT to SDEs of the form Eq. (1). Since both x and w are random quantities, it is more convenient to write out the SDE in the following form

$$\frac{dz(t)}{dt} = F(z(t);\theta) + De(t), \tag{11}$$

where $e(t)$ is a zero-mean white noise process with covariance $Q_c(t)$ and D is a dispersion matrix. We can express Eq. (1) in this form by defining an augmented state $z(t) = [x(t), w(t)]^\top$ and defining $F(\cdot)$ and D as follows

$$F(z(t);\theta) = \begin{bmatrix} f(x(t), g(x(t);\theta)) + h(x(t), g(x(t);\theta))w(t) \\ 0 \end{bmatrix}, \qquad D = \begin{bmatrix} 0 & 0 \\ 0 & I \end{bmatrix}.$$

As shown in [53] (Algorithm 4.4), the predicted mean and covariance for any time $t \geq t_k$ can be computed from the initial mean $m(t_k) = [\mu_x(t_k), 0]^\top$ and covariance $P(t_k) = \mathrm{diag}(\Sigma_x(t_k), I)$ (estimated from data as discussed in the previous section) by integrating the following differential equations

$$\frac{dm(t)}{dt} = F(Z(t);\theta) w_m, \tag{12}$$
$$\frac{dP(t)}{dt} = Z(t)F^\top(Z(t);\theta) + F(Z(t);\theta) W Z^\top(t) + D Q_c(t) D^\top,$$

where the sigma points $Z(t)$ are defined similarly to that in Eq. (9), with $m(t)$ and $P(t)$ now being functions of time. We can then recover the original state mean and covariance by simple transformation of the augmented state

$$\hat{\mu}_x(t|t_k;\theta) = \begin{bmatrix} I & 0 \end{bmatrix} m(t;\theta), \quad \hat{\Sigma}_x(t|t_k;\theta) = \begin{bmatrix} I & 0 \\ 0 & 0 \end{bmatrix} P(t;\theta), \tag{13}$$

where we have used the notation $t|t_k$ to denote predicted quantities given initial information at time t_k.

We use ODE solvers within the neural ODE framework [41,42] to integrate Eq. (12) since $F(\cdot)$ is defined in terms of an embedded neural network used to represent the unknown/hidden physics $g(\cdot)$. The flexible choice of ODE solver provides SPINODE with the ability to accurately handle systems with high-dimensional, stiff, and/or nonlinear dynamics. Another advantage of explicitly integrating the SDE (as opposed to applying a fixed time step discretization) is that we can handle potentially sparse, non-uniform time grids $\{t_0, t_1, \ldots, t_K\}$. Although we have only exploited information provided by the first two moments of the state distribution, UT can also straightforwardly incorporate higher-order moment data, as described in [74,75]. As shown in Section 4.2, incorporating higher-order moments into the prediction scheme can lead to improved performance when learning $g(\cdot)$ due to better placement of the sigma points.

2.3. Moment-matching

Since we represent the hidden physics $g(x;\theta)$ with a neural network, we need to define a proper loss function to estimate θ. In other words, given a loss function $L(\theta)$, we can translate our goal of "learning the hidden physics" into solving the following optimization problem:

$$\theta^\star = \text{argmin}_\theta \; L(\theta). \tag{14}$$

A natural loss function for the moment-matching problem is the reconstruction error of the moments, which can be defined as follows

$$L(\theta) = \sum_{k=1}^{K} \sum_{i=1}^{N_m} \| m_x^{(i)}(t_k) - \hat{m}_x^{(i)}(t_k|t_{k-1};\theta) \|^2, \tag{15}$$

which simplifies to the following expression when only the first two moments are considered

$$L(\theta) = \sum_{k=1}^{K} \| \mu_x(t_k) - \hat{\mu}_x(t_k|t_{k-1};\theta) \|^2 + \| \Sigma_x(t_k) - \hat{\Sigma}_x(t_k|t_{k-1};\theta) \|^2, \tag{16}$$

where $\| \cdot \|^2$ denotes the sum of squared values of all elements in the vector/matrix. We solve Eq. (14) via mini-batch gradient descent, which estimates the gradient of the loss function as follows

$$\nabla_\theta L(\theta) \approx \frac{1}{B} \sum_{k \in \mathcal{B}} \nabla_\theta L_k(\theta), \tag{17}$$

where $L_k(\theta) = \| \mu_x(t_k) - \hat{\mu}_x(t_k|t_{k-1};\theta) \|^2 + \| \Sigma_x(t_k) - \hat{\Sigma}_x(t_k|t_{k-1};\theta) \|^2$ is the error in the k^{th} data point, B is the number of "mini-batch" samples, and $\mathcal{B} \subset \{1, \ldots, K\}$ is a set of B randomly drawn indices. We can efficiently evaluate the gradient estimate in Eq. (17) using the adjoint sensitivity method described in [41,54]. Therefore, SPINODE can be easily implemented using open-source deep learning software such as PyTorch [76] – we have provided an implementation for the case studies considered in this work on GitHub [58].

2.4. Simplified training procedure with approximate unscented transform

Based on the UT-based ODEs in Eq. (12) and the structure of $F(\cdot)$, the evaluation of the mean and covariance are fully coupled, that is, $m(t)$ and $P(t)$ must be simultaneously integrated to evaluate the loss function and its gradient. Since this procedure can be computationally expensive, it is useful to derive alternative approximations that can lead to a simplified training procedure. A particularly important special case of Eq. (1) is when the hidden physics $g(\cdot)$ are fully separable, i.e.,

$$dx = f(x, g_1(x;\theta_1)) + h(x, g_2(x;\theta_2))dw, \tag{18}$$

where $g_1(x;\theta_1)$ and $g_2(x;\theta_2)$ denote two completely independent neural networks (each with their own set of local parameters). According to Eq. (12), $\theta = \{\theta_1, \theta_2\}$ must still be trained simultaneously since the sigma points depend on both the mean and covariance.

To simplify the training process, we present an approximate UT that formulates independent ODEs that describe the time evolution of the transformed sigma points $Y = F(Z)$, where $Z = [Z^x, Z^w]^\top$:

$$\frac{dY}{dt} = f(Z^x(t), g_1(Z^x(t);\theta)) + h(Z^x(t), g(Z^x(t);\theta))Z^w(t). \tag{19}$$

The predictions of Y combined with Eq. (9) can be used to predict the mean and covariance. More importantly, since $Z^w(t)$ has a mean of zero and appears in an additive fashion, as long as the weights are chosen in a symmetric fashion, the $h(Z^x(t), g(Z^x(t);\theta))Z^w(t)$ term will cancel when evaluating the mean of the state. Therefore, in this case, the predicted

state mean only depends on θ_1, i.e., $\hat{\mu}_x(t|t_k;\theta_1)$. Assuming that the predicted state covariance depends weakly on θ_1, we can then separately train θ_1 and θ_2. In particular, we sequentially solve the following two smaller optimization problems:

$$\theta_1^\star = \text{argmin}_{\theta_1} \sum_{k=0}^{K} \|\mu_x(t_k) - \hat{\mu}_x(t_k|t_{k-1};\theta_1)\|^2, \tag{20}$$
$$\theta_2^\star = \text{argmin}_{\theta_2} \sum_{k=0}^{K} \|\Sigma_x(t_k) - \hat{\Sigma}_x(t_k|t_{k-1};\theta_1^\star,\theta_2)\|^2.$$

Note that the second optimization problem above is solved using a fixed functional form for the drift term $g_1(x;\theta_1^\star)$. Although heuristic in nature, this decomposed training strategy greatly reduces the number of parameters that need to be simultaneously considered when evaluating the loss function gradients. Not only does this significantly reduce computational cost, it also limits the search space such that we are less likely to find solutions that result in overfitting.

2.5. Validation criteria using predicted state distribution

It is important to note that there can be many values for parameters θ that result in small or even zero loss function values since moments only provide limited information about the underlying distributions. In other words, even though two different sets of neural network parameters produce the same loss function value, they may result in substantially different predicted state distributions. We can develop a validation test to determine whether or not a given set of optimal parameter values $\theta^\star$ results in accurate state distributions. In particular, we can evaluate the sum of the Kullback–Leibler (KL) divergence [77] between the measured state distributions p_{x_k} and predicted $\hat{p}_{x_k}(\theta^\star)$ state distributions from a given initial condition over time, i.e.,

$$\text{Validation Error} = \sum_{k=0}^{K_V} \int_{\mathcal{X}} p_{x_k}(x) \log\left(\frac{p_{x_k}(x)}{\hat{p}_{x_k}(x;\theta^\star)}\right) dx, \tag{21}$$

where K_V denotes the number of validation time steps. Note that one can easily modify this definition to include multiple initial conditions and other controlled input values. Since we cannot evaluate either of these distributions exactly, we need to rely on established sample-based probability density function estimation techniques such as kernel density estimation [78]. We recommend using this validation error criteria to decide if the hidden physics have been learned accurately enough to make reasonable predictions. Whenever the validation error is large, there may be a need to either modify the training strategy, increase the number of moments considered in the loss function, or collect additional data. Due to its simplicity, it is useful to start with the training procedure described in Section 2.4 and, if it does not pass the validation error test described in this section, apply the more detailed coupled training strategy.

3. Case studies

We demonstrate SPINODE on three benchmark *in-silico* case studies from the literature: (i) a two-state model for directed colloidal self-assembly with an exogenous input [55], (ii) a four-state competitive Lotka-Volterra model with a coexistence equilibrium [56], and (iii) a six-state SIR epidemic model for disease spread [57]. Each of these stochastic dynamical systems can be modeled by Eq. (1), and, since the hidden physics are fully separable in each case, Eq. (18).

State trajectory data is collected by discretizing Eq. (1) according to an Euler-Maruyama discretization scheme [79,80]. These discretized SDEs are meant to represent the "real" system dynamics. Data-estimated moments $m_x^{(i)}(t_k)$ are then collected according to the approach described in Section 2.1 and the SPINODE framework outlined in Sections 2.2–2.4 is used to learn (or reconstruct) the hidden physics $g(x)$ from the collected stochastic trajectory data. SPINODE's performance is evaluated by assessing the accuracy of the reconstructed hidden physics. In each case study, moments $m_x^{(i)}(t_k)$ are calculated from 10^5 replicates of 50 time-step state trajectories from 2000 unique initial conditions (which leads to 10^5 total moments $m_x^{(i)}(t_k)$). As mentioned in Section 2.1, the number of initial conditions could very likely be decreased by employing more advanced sampling strategies, but exploring such strategies is beyond the scope of this work. Section 4.2 examines the relationship between the total number of data points and trajectory replicates and the hidden physics reconstruction accuracy.

3.1. Case study 1: directed colloidal self-assembly with an exogenous input

The first case study is a two-state model for directed colloidal self-assembly with an exogenous input [55]. Here, the voltage of an external electric field is adjusted to mediate the two-dimensional self-assembly of silica micro-particles. The system dynamics are modeled according to Eq. (1). Denote x as an order parameter that represents crystal structure (i.e.,

the system state), u as the electric field voltage (i.e., the exogenous input), K_b as Boltzmann's constant, and T as the temperature:

$$
\begin{aligned}
&dx = g_1(x,u)dt + \sqrt{2g_2(x,u)}dw,\\
&g_1(x,u) = \frac{d}{dx}\Big(g_2(x,u)\Big) - \frac{d}{dx}\Big(F(x,u)\Big)\frac{g_2(x,u)}{K_bT},\\
&g_2(x,u) = 4.5\times 10^{-3}e^{-(x-2.1-0.75u)^2} + 0.5\times 10^{-3},\\
&F(x,u) = 10K_bT(x-2.1-0.75u)^2.
\end{aligned}
\tag{22}
$$

The hidden physics are the drift coefficient, $g_1(x,u)$, and the diffusion coefficient, $g_2(x,u)$. Note that $g_1(x,u)$ is a function of a $g_2(x,u)$ and the free energy landscape $F(x,u)$. This relationship provides an example of how drift and diffusion coefficients can be used to derive other hidden system physics.

We chose this case study because the hidden physics are highly nonlinear and depend on an exogenous input. To our knowledge, no previously reported approach for learning SDE hidden physics has explicitly learned $g(x,u)$. Instead, existing approaches typically seek to learn $g(x)$ at discrete values of u and interpolate [9,10,15,32,55]. This requires repeating the entire hidden physics learning procedure for many discrete values of u and, thus, demands a trade-off between computational cost and accuracy. SPINODE, on the other hand, directly learns $g(x,u)$ over the entire (x,u) state space.

3.2. Case study 2: competitive Lotka-Volterra with a coexistence equilibrium

The second case study is a four-state competitive Lotka-Volterra model with a coexistence equilibrium [56]. The stochastic dynamics are modeled according to Eq. (1). Note that $x = [x_1, x_2]^\top$ and x_i^{eq} are the coexistence equilibrium points:

$$
\begin{aligned}
&dx_1 = g_1(x)_1dt + \sqrt{2g_2(x)_1}dw_1,\\
&dx_2 = g_1(x)_2dt + \sqrt{2g_2(x)_2}dw_2,\\
&g_1(x)_1 = x_1(1-x_1-k_1x_2),\\
&g_1(x)_2 = x_2(1-x_2-k_2x_1),\\
&g_2(x)_1 = x_1(x_2-x_2^{\text{eq}}),\\
&g_2(x)_2 = x_2(x_1-x_1^{\text{eq}}),\\
&x_1^{\text{eq}} = \frac{1-k_1}{1-k_1k_2},\quad x_2^{\text{eq}} = \frac{1-k_2}{1-k_1k_2},\\
&k_1 = 0.4,\quad k_2 = 0.5.
\end{aligned}
\tag{23}
$$

The hidden physics are the two-dimensional drift and diffusion coefficients, $g_1(x_1,x_2)$ and $g_2(x_1,x_2)$. As a result, we seek to train two multi-input, multi-output neural networks that approximate the hidden physics. The drift coefficient neural network takes x_1 and x_2 as input and outputs $g_1(x_1,x_2)_1$ and $g_1(x_1,x_2)_2$. The diffusion coefficient neural network takes x_1 and x_2 as input and outputs $g_2(x_1,x_2)_1$ and $g_2(x_1,x_2)_2$.

We chose this case study because both the drift and diffusion coefficients are multi-dimensional and nonlinear. We apply SPINODE to this "more complex" SDE to demonstrate the framework's scalability. Note that no aspect of the framework was altered from its implementation for the previous case study.

3.3. Case study 3: susceptible-infectious-recovered epidemic model

The third case study is a six-state SIR epidemic model for disease spread [57]. The stochastic dynamics are modeled according to Eq. (1):

$$
\begin{aligned}
&dS = (b - dS - g(S,I) + \gamma R)dt + \sigma_1 S dw_1,\\
&dI = (g(S,I) - (d+\mu+\delta)I)dt + \sigma_2 I dw_2,\\
&dR = (\mu I - (d+\gamma)R)dt + \sigma_3 R dw_3,\\
&g(S,I) = \frac{kS^hI}{S^h + \alpha I^h},\\
&b = 1,\ d = 0.1,\ k = 0.2,\ \alpha = 0.5,\ \gamma = 0.01,\ \mu = 0.05,\\
&\delta = 0.01,\ h = 2,\ \sigma_1 = 0.2,\ \sigma_2 = 0.2,\ \sigma_3 = 0.1.
\end{aligned}
\tag{24}
$$

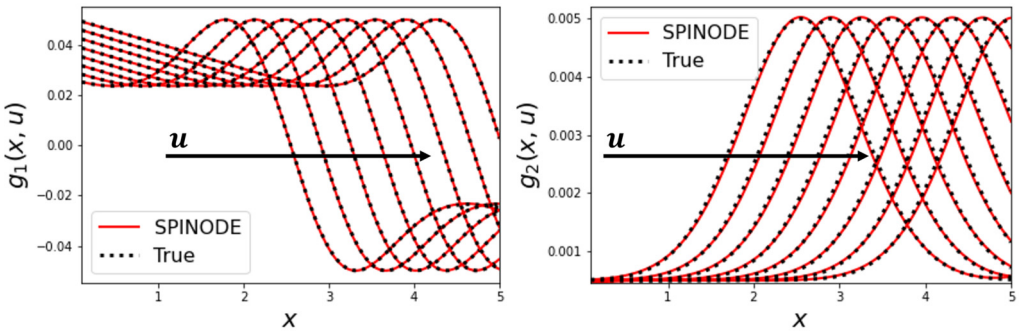

Fig. 2. Learned hidden physics of directed colloidal self-assembly system with an exogenous input. SPINODE learns the drift and diffusion coefficients $g_1(x, u)$ and $g_2(x, u)$ of the stochastic dynamical system described by Eq. (22) with high accuracy.

The hidden physics is the infection transmission rate, $g(S, I)$, which plays a key role in determining disease spread dynamics in many epidemic models [17,18,57,63,81–84]. The form of $g(S, I)$ is widely considered to be unknown, and each of the above-listed references propose different versions of this function. We apply SPINODE to Eq. (24) to learn $g(S, I)$. We chose this case study to demonstrate that SPINODE can not only broadly learn drift and diffusion coefficients but also can learn specific unknown physics terms within complex SDEs.

The hidden physics $g(S, I)$ primarily contribute to the deterministic dynamics (i.e., $f(x, g(x)$ in Eq. (1)) and appears in the time evolution equations for both S and I in Eq. (24). The resulting loss function used to train $g(S, I; \theta)$ is then given by:

$$\min_{\theta} \sum_{k=0}^{K} \left\| \hat{\mu}_S(t_k) - \mu_S(t_k)) \right\|^2 + \left\| \hat{\mu}_I(t_k) - \mu_I(t_k)) \right\|^2, \tag{25}$$

while the loss functions used to train $g_1(x, u)$ and $g_1(x_1, x_2)$ in the previous two case studies were given by Eq. (20).

4. Results and discussion

4.1. Learning hidden physics

We demonstrate SPINODE on the case studies outlined in Section 3. In each case study, moments $m_x^{(i)}(t_k)$ (e.g., means and covariances) are estimated from stochastic trajectory data. The approximate UT method described in Section 2.4 is then used to yield deterministic ODEs that describe the time evolution of the sigma points. An Euler ODE scheme is used to solve these ODEs and thus predict the time evolution of the means and covariances. Mini-batch gradient descent with adjoint sensitivity is then used to (i) match the predicted means and covariances to the data-estimated means and covariances and (ii) train the neural networks $g(x; \theta)$ that approximate the true hidden physics $g(x)$. Note that the time intervals over which the means and covariances are predicted (i.e., the sampling times) are approximately $1/50^{\text{th}}$ of the time it takes each system to reach steady state.

In principle, SPINODE's performance can vary from run-to-run due to the randomness involved in neural network weight initialization and assigning data-estimated moments $m_x^{(i)}(t_k)$ to training, validation, and test sets. We thus assessed SPINODE's performance by calculating the root mean squared errors (RMSE) between the learned hidden physics $g(x; \theta^*)$ and the actual system hidden physics $g(x)$ over 30 SPINODE runs with randomly selected initial weight values and training/validation/test set data assignments. Table 1 shows the mean and standard deviations of these RMSEs while Figs. 2 – 4 show a visual comparison of $g(x; \theta^*)$ and $g(x)$ for representative runs. In each case, SPINODE learns the hidden physics $g(x; \theta^*)$ with high accuracy and low run-to-run variation. We note that in the real-world, the actual values of the hidden physics $g(x)$ will be unavailable. In these cases, SPINODE's performance should be validated via the methodology described in Section 2.5., i.e., by comparing the data-estimated moments of trajectories from the real dynamics $m_x^{(i)}(t_k)$ to those generated from the learned dynamics involving $f(\cdot)$, $h(\cdot)$, and $g(x, \theta^*)$. Visual representations of the time-evolution of the probability distributions of the states from randomly selected initial conditions and exogenous input values for the colloidal self-assembly and Lotka-Volterra case studies are shown in Figs. 5 – 6.

We further note that the hidden physics reconstructions shown in Figs. 2 – 4 and Table 1 essentially occur under "ideal" conditions – as $g(x; \theta)$ is trained using a large number of moments $m_x^{(ij)}(t_k)$ that are estimated from a large number of repeated trajectories from a large number of initial conditions (see Section 2.1 and Section 3 for details). In addition, the sampling times are identical to the discretization times used in the Euler-Maruyama simulations that represent the "true" system dynamics. This last point motivated the use of an Euler ODE solver to predict the moment time-evolution. In the next section, we assess the performance of SPINODE in terms of decreasing the number of repeated trajectories used to estimate

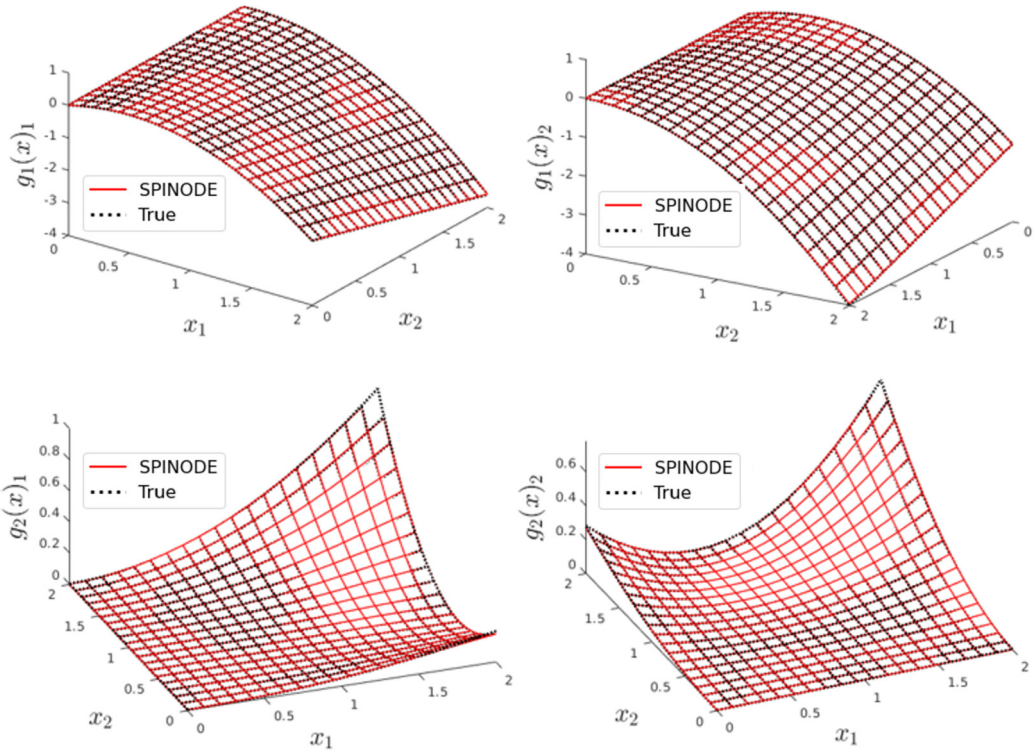

Fig. 3. Learned hidden physics of competitive Lotka-Volterra with a coexistence equilibrium. SPINODE learns the drift and diffusion coefficients $g_1(x_1, x_2)_{1,2}$ and $g_2(x_1, x_2)_{1,2}$ of the stochastic dynamical system described by Eq. (23) with high accuracy.

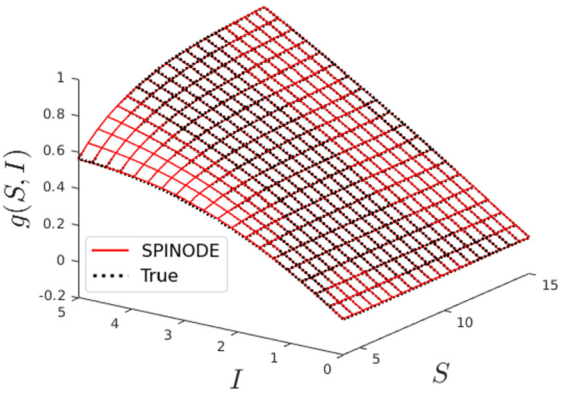

Fig. 4. Learned hidden physics of susceptible-infectious-recovered (SIR) epidemic model. SPINODE learns $g(S, I)$ from Eq. (24) with high accuracy.

the moments $m_x^{(ij)}(t_k)$, decreasing the total number of moments $m_x^{(ij)}(t_k)$ used to train $g(x;\theta)$, altering the uncertainty propagation strategy, and adjusting the sampling time.

4.2. Numerical robustness

Figs. 7-8 show RMSEs between the learned hidden physics $g(x;\theta^*)$ and the actual system hidden physics $g(x)$ for the Lotka-Volterra and SIR epidemic case studies as a function of the total number of repeated trajectories used to calculate the data-estimated moments $m_x^{(i)}(t_k)$. As can be seen, the RMSEs converge between 10^3 and 10^4 total repeats in both case studies and the RMSEs grow very quickly under 10^2 total repeats. Figs. 7-8 highlight that SPINODE's ability to the learn the hidden physics $g(x)$ critically hinges on how accurately moments $m_x^{(ij)}(t_k)$ can be estimated from data. In this work, moments are estimated from data by repeating (many) stochastic trajectories from identical initial conditions. Section 2.1 discusses how this strategy is not appropriate for systems in which one does not have control over initial conditions, number of replica runs, consistent measurement times, etc. Section 2.1 also suggests potential methods for learning data-estimated moments in such cases.

Table 1
Reconstruction root mean square errors of learned hidden physics. SPINODE is used to learn the hidden physics of the case studies in Sections 3.1 (directed colloidal self-assembly with an exogenous input), 3.2 (competitive Lotka-Volterra with a coexistence equilibrium), and 3.3 (susceptible-infectious-recovered epidemic model). The root mean square error (RMSE) between the learned hidden physics $g(x;\theta^*)$ and actual hidden physics $g(x)$ is then calculated. This process is repeated 30 times with randomly selected initial weight values and training/validation/test set data assignments. The means and standard deviations (std) of the calculated RMSEs are shown. For each case study, SPINODE learns the hidden physics $g(x;\theta^*)$ with high accuracy and low run-to-run variation.

Case Study	RMSE Mean	RMSE Std
Colloidal Self-Assembly, $g_1(x,u)$	1.33×10^{-4}	2.77×10^{-5}
Colloidal Self-Assembly, $g_2(x,u)$	4.97×10^{-5}	4.67×10^{-6}
Lotka-Volterra , $g_1(x_1,x_2)_1$	9.20×10^{-4}	1.24×10^{-4}
Lotka-Volterra , $g_1(x_1,x_2)_2$	7.91×10^{-4}	7.59×10^{-5}
Lotka-Volterra , $g_2(x_1,x_2)_1$	3.93×10^{-3}	5.97×10^{-5}
Lotka-Volterra , $g_2(x_1,x_2)_2$	4.88×10^{-3}	7.15×10^{-5}
Susceptible-Infectious-Recovered , $g(S,I)$	2.62×10^{-3}	1.89×10^{-4}

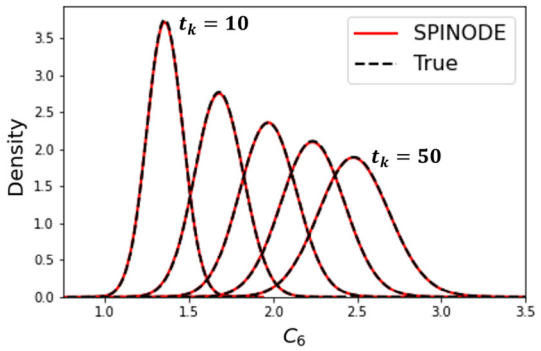

Fig. 5. Time evolution of kernel density estimates for directed colloidal self-assembly system with an exogenous input. Trained neural networks $g_1(x,u;\theta_1^\star)$ and $g_2(x,u;\theta_2^\star)$ are used to simulate the system dynamics from a randomly selected initial condition with a randomly selected exogenous input. The true dynamics are then simulated using the same initial condition and exogenous input. In each case, the stochastic trajectory is repeated 10^5 times and kernel density functions are calculated at each sampling time. Estimates of the kernel density function for the "true" and "learned" dynamics at select sampling times are plotted against one another. SPINODE reproduces the kernel density function with high accuracy.

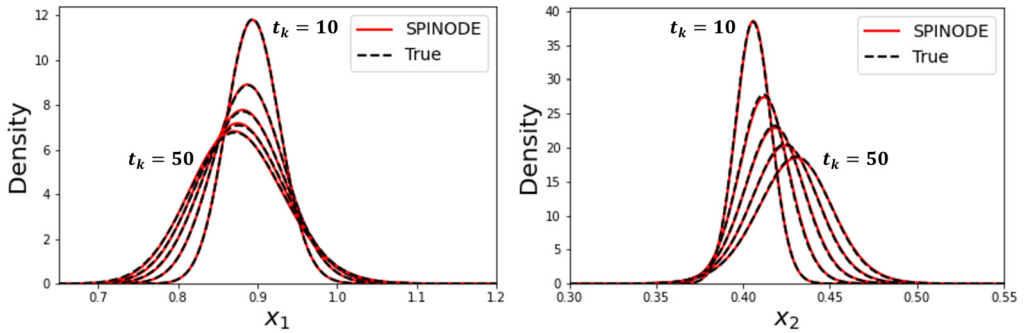

Fig. 6. Time evolution of kernel density estimates for Lotka-Volterra with a coexistence equilibrium. Trained neural networks $g_1(x_1,x_2;\theta_1^\star)$ and $g_2(x_1,x_2;\theta_1^\star)$ are used to simulate the system dynamics from a randomly selected initial condition. The true dynamics are then simulated using the same initial condition. In each case, the stochastic trajectory is repeated 10^5 times and kernel density functions are calculated at each sampling time. Estimates of the kernel density function for the "true" and "learned" dynamics at select sampling times are plotted against one another. SPINODE reproduces the kernel density function with high accuracy.

Figs. 9-10 plot the RMSEs between the learned hidden physics $g(x;\theta^*)$ and the actual system hidden physics $g(x)$ for the Lotka-Volterra and SIR epidemic case studies as a function of the total number of data-estimated moments $m_x^{(i)}(t_k)$ used to train $g(x;\theta)$. In this case, each moment $m_x^{(i)}(t_k)$ is estimated using 10^5 total repeated trajectories – only the total number of moments used to train $g(x;\theta)$ is varied. Both figures show that more training data can lead to a more accurate recovery of the hidden physics. The amount of training data required for the RMSEs to converge depends on a combination of the complexity of $g(x)$ and the "informativeness" of the loss function used to train $g(x;\theta)$. For example, the behavior of $g(S,I)$ in SIR epidemic case study can be considered more nonlinear than that of $g_1(x_1,x_2)_1$ and $g_1(x_1,x_2)_2$ in the Lotka-Volterra case study, which is more nonlinear still than that of $g_2(x_1,x_2)_1$ and $g_2(x_1,x_2)_2$ in the Lotka-Volterra case study. Correspondingly, the RMSEs of $g_2(x_1,x_2)_1$ and $g_2(x_1,x_2)_2$ converge after fewer total data points than the other

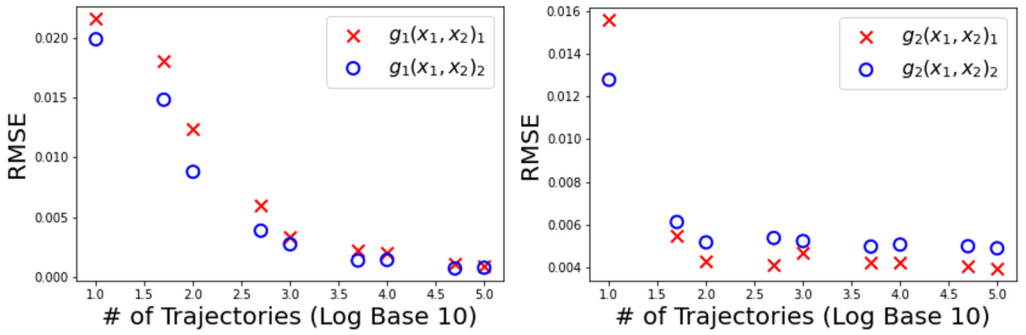

Fig. 7. Sampling sensitivity analysis: competitive Lotka-Volterra with a coexistence equilibrium. SPINODE trains the neural networks that approximate $g_1(x_1, x_2)_{1,2}$ and $g_2(x_1, x_2)_{1,2}$ after decreasing the total number of repeated stochastic trajectories used to estimate the moments $m_x^{(i)}(t_k)$ from data. The root mean square errors (RMSEs) between the learned and actual hidden physics are then calculated. The RMSEs converge around 10^3 total repeats for $g_1(x_1, x_2)_{1,2}$ and 10^2 total repeats for $g_2(x_1, x_2)_{1,2}$.

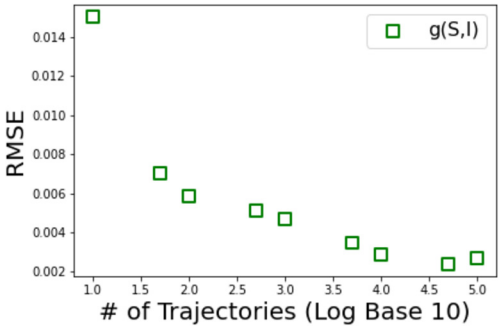

Fig. 8. Sampling sensitivity analysis: susceptible-infectious-recovered (SIR) epidemic model. SPINODE trains the neural network that approximates $g(S, I)$ after decreasing the total number of repeated stochastic trajectories used to estimate the moments $m_x^{(i)}(t_k)$ from data. The root mean square error (RMSE) between the learned and actual hidden physics is then calculated. The RMSE converges around 10^4 total repeats.

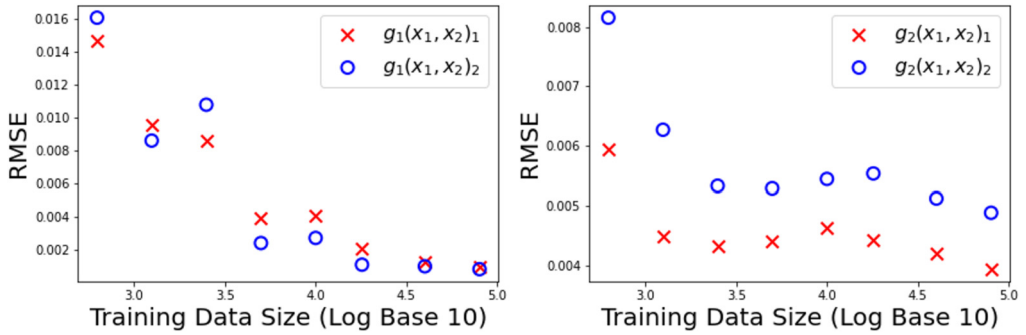

Fig. 9. Training data size sensitivity analysis: competitive Lotka-Volterra with a coexistence equilibrium. SPINODE trains the neural networks that approximate $g_1(x_1, x_2)_{1,2}$ and $g_2(x_1, x_2)_{1,2}$ after decreasing the size of the training data (i.e., the total number of data-estimated moments $m_x^{(i)}(t_k)$). The root mean square error (RMSE) between the learned and actual hidden physics is then calculated. The RMSEs converge around 2.5×10^4 total moments for $g_1(x_1, x_2)_{1,2}$ and 5×10^3 total moments for $g_2(x_1, x_2)_{1,2}$.

hidden physics terms. Despite the more nonlinear behavior of $g(S, I)$, however, its RMSE converges earlier than the RMSEs of $g_1(x_1, x_2)_1$ and $g_1(x_1, x_2)_2$. We note that the cost function used to train $g(S, I)$ is more "informative" than the cost functions used to train $g_1(x_1, x_2)$ and $g_2(x_1, x_2)$ – compare Eq. (20) to Eq. (25)) – as Eq. (25) contains added information from multiple known "physical" terms in Eq. (24). Overall, the general notion that more training data can lead to higher-performing neural network models is expected [85]. However, the fact that $g(S, I)$'s RMSE seems to converge at fewer total data points suggests the previously reported observation [43] that incorporating more physics into the cost function can reduce data requirements for training neural networks.

We next use the colloidal self-assembly case study to investigate SPINODE's sensitivity to the chosen uncertainty propagation method. Fig. 11 shows SPINODE's reconstruction of the hidden physics $g_1(x, u)$ and $g_2(x, u)$ when propagating stochasticity via linearization [73] and two methods based on unscented transform – UT-2M and UT-4M. UT-2M, which is

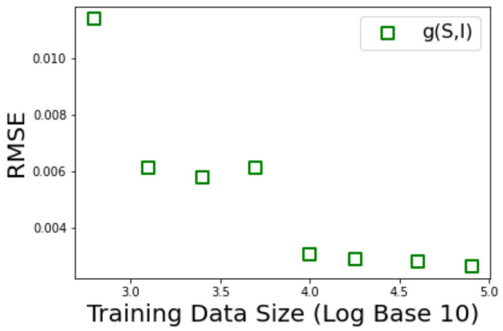

Fig. 10. Training data size sensitivity analysis: susceptible-infectious-recovered (SIR) epidemic model. SPINODE trains the neural networks that approximate $g(S, I)$ after decreasing the size of the training data (i.e., the total number of data-estimated moments $m_x^{(i)}(t_k)$). The root mean square error (RMSE) between the learned and actual hidden physics is then calculated. The RMSE converges slightly before 10^4 total moments.

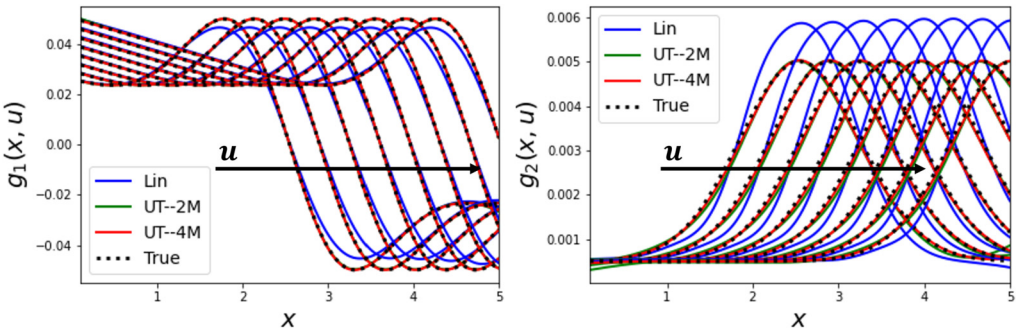

Fig. 11. Uncertainty propagation sensitivity analysis: directed colloidal self-assembly with an exogenous input. SPINODE trains neural networks that approximate $g_1(x, u)$ and $g_2(x, u)$ using three different uncertainty propagation methods: linearization (Lin), unscented transform with two moments (UT-2M) and unscented transform with four moments (UT-4M). UT-2M describes the time evolution of the mean and covariance based on the data-estimated mean and covariance at previous time points while UT-4M describes the time evolution of the mean and covariance based on the data-estimated means, covariance, skew, and kurtosis at previous time points. SPINODE with UT-2M and UT-4M significantly outperforms SPINODE with linearization, while SPINODE with UT-4M slightly outperforms SPINODE with UT-2M for learning $g_2(x, u)$.

explained in detail in Sections 2.2 and 2.4, describes the time evolution of the mean and covariance based on the data-estimated mean and covariance at previous time points. UT-4M, which can be viewed as an extension of UT-2M based on the work in [74], describes the time evolution of the mean and covariance based on the data-estimated means, covariance, skew, and kurtosis at previous time points. SPINODE with both UT methods significantly outperforms SPINODE with linearization. This performance discrepancy indicates that the UT methods propagate stochasticity through Eq. (22) much more accurately than the linearization method does. While SPINODE with UT-2M and UT-4M learn $g_1(x, u)$ with near identical accuracy, the RMSE of SPINODE with UT-4M's recovery of $g_2(x, u)$ is marginally lower than the RMSE of SPINODE with UT-2M's recovery of $g_2(x, u)$ (i.e., 8.64×10^{-5} vs. 4.67×10^{-5}). UT-4M thus leads to a more accurate prediction of the time evolution of the covariance than UT-2M does, as only the covariance is used to train $g_2(x, u)$ (see Eq. (20)). The latter point supports our earlier remark that SPINODE's ability to incorporate higher moments can make SPINODE well-suited for learning $g(x)$ when $f(x, (g(x))$ and $h(x, (g(x))$ are highly nonlinear and the distribution of x is non-Gaussian as a result. We note that although kernel density estimations in Fig. 5 appear fairly Gaussian, the relatively minor skews and kurtoses of the distributions of $x(t)$ are still large enough to affect the uncertainty propagation.

We further investigate SPINODE's sensitivity to uncertainty propagation by extending the sampling times at which data-estimated moments are collected. All previous results for the colloidal self-assembly case study used a sampling time of 1 second. Fig. 12a plots the RMSEs of $g_1(x, u)$ and $g_2(x, u)$ (when UT-4M is implemented for uncertainty propagation) as a function of sampling time. The RMSE increases nearly linearly with sampling time. Fig. 12b shows that the prediction errors for the mean and covariance also increase nearly linearly with sampling time. It is thus reasonable to suggest that SPINODE's sensitivity to sampling time in the colloidal self-assembly case study can be attributed to the sensitivity of the uncertainty propagation method to the sampling time.

Above all else, Figs. 11,12 demonstrate SPINODE's sensitivity to the choice of uncertainty propagation method. Section 1 discusses how SPINODE can in principle accommodate any uncertainty propagation method. As a result, the uncertainty propagation method should be viewed as a "hyper-parameter" within SPINODE. We finally discuss the choice of the ODE solver within SPINODE. Because the case study simulation data was generated via an Euler-Maruyama discretization, an Euler ODE solver within SPINODE yielded the most accurate reconstructions of the hidden physics. The current implementation of SPINODE [58], however, includes other advanced, even adaptive time-step solvers that have been shown to accurately

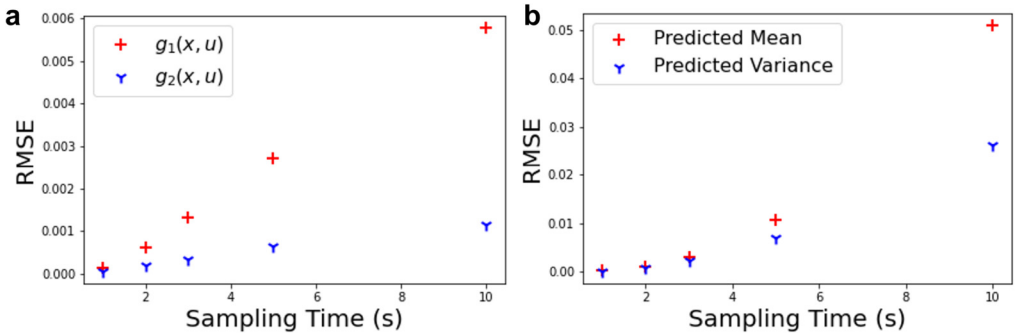

Fig. 12. Sampling time sensitivity analysis: directed colloidal self-assembly with an exogenous input. (a) SPINODE trains neural networks that approximate $g_1(x,u)$ and $g_2(x,u)$ using the UT-4M uncertainty propagation method with different sampling times. The root mean square error (RMSE) between the learned and actual hidden physics is then calculated. (b) UT-4M is used to propagate stochasticity through the true dynamics (i.e., Eq. (22)) for each of the data-estimated moments in the training data set at different sampling times. The RMSEs between the predicted and the data-estimated means and covariances at the next sampling time are then calculated. The errors in reconstructing the hidden physics and predicting the mean and covariance grow nearly linearly with the sampling time.

integrate high-dimensional, stiff, and nonlinear ODEs [41]. The choice of ODE solver should thus also be viewed as a "hyper-parameter" within SPINODE. In fact, known hyper-parameter optimization strategies such as Bayesian Optimization [86] can be used to determine the "best" ODE solver to use during training.

5. Conclusions and future work

We proposed a flexible and scalable framework based on the notions of neural ordinary differential equations, physics-informed neural networks, and moment-matching for training deep neural networks to learn constitutive equations that represent hidden physics within stochastic differential equations. We demonstrated the proposed stochastic physics-informed neural ordinary differential equation framework on three benchmark *in-silico* case studies from the literature. We analyzed the performance of the proposed framework in terms of its repeatability, sensitivity to weight initialization and training/validation/testing set allocation, total number of data points, total number of repeated trajectories, uncertainty propagation method, and sampling time. We showed the framework's scalability by learning highly nonlinear hidden physics within multidimensional stochastic differential equations with multiplicative noise. We illustrated the framework's flexibility by (i) learning both general drift and diffusion coefficients (with or without an exogenous input) and specific unknown functions within stochastic differential equations for different systems and (ii) demonstrating that key aspects of the framework (e.g., the choice of uncertainty propagation method) can be easily and independently adjusted. An open challenge is the fact that a large number of repeated state trajectories are required to accurately learn hidden physics. We will focus future work on learning probability distributions directly from data instead of estimating moments from repeated stochastic trajectories from identical initial conditions. To this end, we will explore variational autoencoders [64–66], generative adversarial networks [45,67–69], and energy-based models [70–72]. Other open challenges include optimizing the uncertainty propagation method and choice of ODE solver during neural network training. We will explore optimizing these hyper-parameter choices using methods based on Bayesian optimization [86]. We will finally explore updating the current implementation of the neural ODE framework based on [41] with more recent and advanced neural ODE framework implementations (e.g., [42]).

CRediT authorship contribution statement

JOL, JAP and AM conceived the idea of the work. JOL performed all the simulations. JOL, JAP and AM wrote the paper.

Declaration of competing interest

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

Acknowledgement

This work is in part supported by the National Science Foundation under Grant 2112754.

References

[1] J. Honerkamp, Stochastic Dynamical Systems: Concepts, Numerical Methods, Data Analysis, John Wiley & Sons, 1993.
[2] N.G. Van Kampen, Stochastic differential equations, Phys. Rep. 24 (3) (1976) 171–228.

[3] N.G. Van Kampen, Stochastic Processes in Physics and Chemistry, vol. 1, Elsevier, 1992.
[4] G. Volpe, J. Wehr, Effective drifts in dynamical systems with multiplicative noise: a review of recent progress, Rep. Prog. Phys. 79 (5) (2016) 053901.
[5] L. Arnold, Stochastic differential equations, New York, 1974.
[6] S. Schuster, M. Marhl, T. Höfer, Modelling of simple and complex calcium oscillations: from single-cell responses to intercellular signalling, Eur. J. Biochem. 269 (5) (2002) 1333–1355.
[7] M. Pereyra, P. Schniter, E. Chouzenoux, J.-C. Pesquet, J.-Y. Tourneret, A.O. Hero, S. McLaughlin, A survey of stochastic simulation and optimization methods in signal processing, IEEE J. Sel. Top. Signal Process. 10 (2) (2015) 224–241.
[8] P.C. Young, Stochastic, dynamic modelling and signal processing: time variable and state dependent parameter estimation, in: Nonlinear and Nonstationary Signal Processing, 2000, pp. 74–114.
[9] X. Tang, B. Rupp, Y. Yang, T.D. Edwards, M.A. Grover, M.A. Bevan, Optimal feedback controlled assembly of perfect crystals, ACS Nano 10 (7) (2016) 6791–6798.
[10] M.A. Bevan, D.M. Ford, M.A. Grover, B. Shapiro, D. Maroudas, Y. Yang, R. Thyagarajan, X. Tang, R.M. Sehgal, Controlling assembly of colloidal particles into structured objects: basic strategy and a case study, J. Process Control 27 (2015) 64–75.
[11] K. Singer, Application of the theory of stochastic processes to the study of irreproducible chemical reactions and nucleation processes, J. R. Stat. Soc., Ser. B, Methodol. 15 (1) (1953) 92–106.
[12] O. Penrose, Nucleation and droplet growth as a stochastic process, in: Analysis and Stochastics of Growth Processes and Interface Models, 2008, p. 265.
[13] C. Zhu, G. Yin, On competitive Lotka–Volterra model in random environments, J. Math. Anal. Appl. 357 (1) (2009) 154–170.
[14] D.H. Nguyen, G. Yin, Coexistence and exclusion of stochastic competitive Lotka–Volterra models, J. Differ. Equ. 262 (3) (2017) 1192–1225.
[15] D.J. Beltran-Villegas, R.M. Sehgal, D. Maroudas, D.M. Ford, M.A. Bevan, Colloidal cluster crystallization dynamics, J. Chem. Phys. 137 (13) (2012) 134901.
[16] G. Hummer, Position-dependent diffusion coefficients and free energies from Bayesian analysis of equilibrium and replica molecular dynamics simulations, New J. Phys. 7 (1) (2005) 34.
[17] C. Bain, Applied mathematical ecology, J. Epidemiol. Community Health 44 (3) (1990) 254.
[18] A. Korobeinikov, P.K. Maini, Non-linear incidence and stability of infectious disease models, Math. Med. Biol. 22 (2) (2005) 113–128.
[19] R. Friedrich, S. Siegert, J. Peinke, M. Siefert, M. Lindemann, J. Raethjen, G. Deuschl, G. Pfister, et al., Extracting model equations from experimental data, Phys. Lett. A 271 (3) (2000) 217–222.
[20] S. Siegert, R. Friedrich, J. Peinke, Analysis of data sets of stochastic systems, Phys. Lett. A 243 (5–6) (1998) 275–280.
[21] R. Friedrich, C. Renner, M. Siefert, J. Peinke, Comment on "indispensable finite time corrections for Fokker-Planck equations from time series data", Phys. Rev. Lett. 89 (14) (2002) 149401.
[22] M. Ragwitz, H. Kantz, Indispensable finite time corrections for Fokker-Planck equations from time series data, Phys. Rev. Lett. 87 (25) (2001) 254501.
[23] D. Lamouroux, K. Lehnertz, Kernel-based regression of drift and diffusion coefficients of stochastic processes, Phys. Lett. A 373 (39) (2009) 3507–3512.
[24] J. Gottschall, J. Peinke, On the definition and handling of different drift and diffusion estimates, New J. Phys. 10 (8) (2008) 083034.
[25] D. Kleinhans, R. Friedrich, A. Nawroth, J. Peinke, An iterative procedure for the estimation of drift and diffusion coefficients of Langevin processes, Phys. Lett. A 346 (1–3) (2005) 42–46.
[26] J. Gradišek, S. Siegert, R. Friedrich, I. Grabec, Analysis of time series from stochastic processes, Phys. Rev. E 62 (3) (2000) 3146.
[27] R. Hegger, G. Stock, Multidimensional Langevin modeling of biomolecular dynamics, J. Chem. Phys. 130 (3) (2009) 034106.
[28] J. Prusseit, K. Lehnertz, Measuring interdependences in dissipative dynamical systems with estimated Fokker-Planck coefficients, Phys. Rev. E 77 (4) (2008) 041914.
[29] A.M. van Mourik, A. Daffertshofer, P.J. Beek, Estimating Kramers–Moyal coefficients in short and non-stationary data sets, Phys. Lett. A 351 (1–2) (2006) 13–17.
[30] D.I. Kopelevich, A.Z. Panagiotopoulos, I.G. Kevrekidis, Coarse-grained kinetic computations for rare events: application to micelle formation, J. Chem. Phys. 122 (4) (2005) 044908.
[31] T.B. Woolf, B. Roux, Molecular dynamics simulation of the gramicidin channel in a phospholipid bilayer, Proc. Natl. Acad. Sci. 91 (24) (1994) 11631–11635.
[32] D.J. Beltran-Villegas, R.M. Sehgal, D. Maroudas, D.M. Ford, M.A. Bevan, A Smoluchowski model of crystallization dynamics of small colloidal clusters, J. Chem. Phys. 135 (15) (2011) 154506.
[33] J. Mittal, T.M. Truskett, J.R. Errington, G. Hummer, Layering and position-dependent diffusive dynamics of confined fluids, Phys. Rev. Lett. 100 (14) (2008) 145901.
[34] J. Mittal, G. Hummer, Pair diffusion, hydrodynamic interactions, and available volume in dense fluids, J. Chem. Phys. 137 (3) (2012) 034110.
[35] A. Ghysels, R.M. Venable, R.W. Pastor, G. Hummer, Position-dependent diffusion tensors in anisotropic media from simulation: oxygen transport in and through membranes, J. Chem. Theory Comput. 13 (6) (2017) 2962–2976.
[36] H. Karimi, K.B. McAuley, Bayesian objective functions for estimating parameters in nonlinear stochastic differential equation models with limited data, Ind. Eng. Chem. Res. 57 (27) (2018) 8946–8961.
[37] D. Bicout, A. Szabo, Electron transfer reaction dynamics in non-Debye solvents, J. Chem. Phys. 109 (6) (1998) 2325–2338.
[38] A. Shrestha, A. Mahmood, Review of deep learning algorithms and architectures, IEEE Access 7 (2019) 53040–53065.
[39] R. Zhang, W. Li, T. Mo, Review of deep learning, arXiv preprint, arXiv:1804.01653, 2018.
[40] H.I. Fawaz, G. Forestier, J. Weber, L. Idoumghar, P.-A. Muller, Deep learning for time series classification: a review, Data Min. Knowl. Discov. 33 (4) (2019) 917–963.
[41] R.T. Chen, Y. Rubanova, J. Bettencourt, D.K. Duvenaud, Neural ordinary differential equations, Adv. Neural Inf. Process. Syst. 31 (2018).
[42] C. Rackauckas, Y. Ma, J. Martensen, C. Warner, K. Zubov, R. Supekar, D. Skinner, A. Ramadhan, A. Edelman, Universal differential equations for scientific machine learning, arXiv preprint, arXiv:2001.04385, 2020.
[43] M. Raissi, P. Perdikaris, 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. 378 (2019) 686–707.
[44] M. Raissi, A. Yazdani, G.E. Karniadakis, Hidden fluid mechanics: learning velocity and pressure fields from flow visualizations, Science 367 (6481) (2020) 1026–1030.
[45] Y. Yang, P. Perdikaris, Adversarial uncertainty quantification in physics-informed neural networks, J. Comput. Phys. 394 (2019) 136–152.
[46] L. Yang, X. Meng, G.E. Karniadakis, B-PINNs: Bayesian physics-informed neural networks for forward and inverse PDE problems with noisy data, J. Comput. Phys. 425 (2021) 109913.
[47] D. Zhang, L. Lu, L. Guo, G.E. Karniadakis, Quantifying total uncertainty in physics-informed neural networks for solving forward and inverse stochastic problems, J. Comput. Phys. 397 (2019) 108850.
[48] A.R. Hall, et al., Generalized Method of Moments, Oxford University Press, 2005.
[49] L. Mátyás, C. Gourieroux, P.C. Phillips, et al., Generalized Method of Moments Estimation, vol. 5, Cambridge University Press, 1999.
[50] J.M. Wooldridge, Applications of generalized method of moments estimation, J. Econ. Perspect. 15 (4) (2001) 87–100.
[51] E.A. Wan, R. Van Der Merwe, The unscented Kalman filter for nonlinear estimation, in: Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium (Cat. No. 00EX373), IEEE, 2000, pp. 153–158.

[52] S.J. Julier, J.K. Uhlmann, New Extension of the Kalman Filter to Nonlinear Systems, Signal Processing, Sensor Fusion, and Target Recognition VI, vol. 3068, International Society for Optics and Photonics, 1997, pp. 182–193.
[53] S. Sarkka, On unscented Kalman filtering for state estimation of continuous-time nonlinear systems, IEEE Trans. Autom. Control 52 (9) (2007) 1631–1641.
[54] L.S. Pontryagin, Mathematical Theory of Optimal Processes, CRC Press, 1987.
[55] X. Tang, Y. Xue, M.A. Grover, Colloidal self-assembly with model predictive control, in: 2013 American Control Conference, IEEE, 2013, pp. 4228–4233.
[56] J. Xiong, X. Li, H. Wang, The survival analysis of a stochastic Lotka-Volterra competition model with a coexistence equilibrium, Math. Biosci. 1 (k1k2) (2019), 1–k1k2.
[57] Y. Cai, Y. Kang, W. Wang, A stochastic SIRS epidemic model with nonlinear incidence rate, Appl. Math. Comput. 305 (2017) 221–240.
[58] J. O'Leary, Stochastic physics-informed neural networks, https://github.com/jtoleary/SPINODE, 2022. (Accessed 1 May 2022), online.
[59] B. Peeters, G. De Roeck, Stochastic system identification for operational modal analysis: a review, J. Dyn. Syst. Meas. Control 123 (4) (2001) 659–667.
[60] E. Reynders, System identification methods for (operational) modal analysis: review and comparison, Arch. Comput. Methods Eng. 19 (1) (2012) 51–124.
[61] X. Hong, R.J. Mitchell, S. Chen, C.J. Harris, K. Li, G.W. Irwin, Model selection approaches for non-linear system identification: a review, Int. J. Syst. Sci. 39 (10) (2008) 925–946.
[62] J. Schoukens, L. Ljung, Nonlinear system identification: a user-oriented road map, IEEE Control Syst. Mag. 39 (6) (2019) 28–99.
[63] I. Bauer, H.G. Bock, S. Körkel, J.P. Schlöder, Numerical methods for optimum experimental design in DAE systems, J. Comput. Appl. Math. 120 (1–2) (2000) 1–25.
[64] D.P. Kingma, M. Welling, Auto-encoding variational Bayes, arXiv preprint, arXiv:1312.6114, 2013.
[65] J. An, S. Cho, Variational autoencoder based anomaly detection using reconstruction probability, Spec. Lect. IE 2 (1) (2015) 1–18.
[66] C. Doersch, Tutorial on variational autoencoders, arXiv preprint, arXiv:1606.05908, 2016.
[67] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, Generative adversarial nets, Adv. Neural Inf. Process. Syst. 27 (2014).
[68] C. Zoufal, A. Lucchi, S. Woerner, Quantum generative adversarial networks for learning and loading random distributions, npj Quantum Inf. 5 (1) (2019) 1–9.
[69] L. Mescheder, S. Nowozin, A. Geiger, Adversarial variational Bayes: unifying variational autoencoders and generative adversarial networks, in: International Conference on Machine Learning, PMLR, 2017, pp. 2391–2400.
[70] Y. LeCun, S. Chopra, R. Hadsell, M. Ranzato, F. Huang, A tutorial on energy-based learning, in: Predicting Structured Data, 2006.
[71] T. Kim, Y. Bengio, Deep directed generative models with energy-based probability estimation, arXiv preprint, arXiv:1606.03439, 2016.
[72] F.K. Gustafsson, M. Danelljan, G. Bhat, T.B. Schön, Energy-based models for deep probabilistic regression, in: European Conference on Computer Vision, Springer, 2020, pp. 325–343.
[73] J.A. Paulson, M. Martin-Casas, A. Mesbah, Input design for online fault diagnosis of nonlinear systems with stochastic uncertainty, Ind. Eng. Chem. Res. 56 (34) (2017) 9593–9605.
[74] K. Ponomareva, P. Date, Z. Wang, A new unscented Kalman filter with higher order moment-matching, in: Proceedings of Mathematical Theory of Networks and Systems (MTNS 2010), Budapest, 2010.
[75] D. Ebeigbe, T. Berry, M.M. Norton, A.J. Whalen, D. Simon, T. Sauer, S.J. Schiff, A generalized unscented transformation for probability distributions, arXiv preprint, arXiv:2104.01958, 2021.
[76] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, S. Chintala, Pytorch: An Imperative Style, High-Performance Deep Learning Library, Advances in Neural Information Processing Systems, vol. 32, Curran Associates, Inc., 2019, pp. 8024–8035.
[77] J.M. Joyce, Kullback-Leibler divergence, in: International Encyclopedia of Statistical Science, Springer, 2011, pp. 720–722.
[78] Y.-C. Chen, A tutorial on kernel density estimation and recent advances, Biostat. Epidemiol. 1 (1) (2017) 161–187.
[79] J.M. Sancho, M. San Miguel, S. Katz, J. Gunton, Analytical and numerical studies of multiplicative noise, Phys. Rev. A 26 (3) (1982) 1589.
[80] P.E. Kloeden, E. Platen, Stochastic differential equations, in: Numerical Solution of Stochastic Differential Equations, Springer, 1992, pp. 103–160.
[81] H.W. Hethcote, The mathematics of infectious diseases, SIAM Rev. 42 (4) (2000) 599–653.
[82] V. Capasso, G. Serio, A generalization of the Kermack-McKendrick deterministic epidemic model, Math. Biosci. 42 (1–2) (1978) 43–61.
[83] W.-m. Liu, S.A. Levin, Y. Iwasa, Influence of nonlinear incidence rates upon the behavior of sirs epidemiological models, J. Math. Biol. 23 (2) (1986) 187–204.
[84] S. Yuan, B. Li, Global dynamics of an epidemic model with a ratio-dependent nonlinear incidence rate, Discrete Dyn. Nat. Soc. (2009) 2009.
[85] G. Hinton, N. Srivastava, K. Swersky, Neural networks for machine learning lecture 6a overview of mini-batch gradient descent, Cited on 14(8) (2012), p. 2.
[86] J. Snoek, H. Larochelle, R.P. Adams, Practical Bayesian optimization of machine learning algorithms, Adv. Neural Inf. Process. Syst. 25 (2012).