

PROCEEDINGS A

rspa.royalsocietypublishing.org

Research

Check for
updates

Article submitted to journal

Subject Areas:

Scientific machine learning, Applied Mathematics, Bayesian statistics, Data-driven modeling

Keywords:

Machine learning, Dynamical systems, Uncertainty quantification, Model discovery

Author for correspondence:

Paris Perdikaris

e-mail: pgp@seas.upenn.edu

Bayesian differential programming for robust systems identification under uncertainty

Yibo Yang¹, Mohamed Aziz Bhouri¹ and Paris Perdikaris¹

¹ Department of Mechanical Engineering and Applied Mechanics
University of Pennsylvania
Philadelphia, PA 19104

This paper presents a machine learning framework for Bayesian systems identification from noisy, sparse and irregular observations of nonlinear dynamical systems. The proposed method takes advantage of recent developments in differentiable programming to propagate gradient information through ordinary differential equation solvers and perform Bayesian inference with respect to unknown model parameters using Hamiltonian Monte Carlo sampling. This allows an efficient inference of the posterior distributions over plausible models with quantified uncertainty, while the use of sparsity-promoting priors enables the discovery of interpretable and parsimonious representations for the underlying latent dynamics. A series of numerical studies is presented to demonstrate the effectiveness of the proposed methods including nonlinear oscillators, predator-prey systems and examples from systems biology. Taken all together, our findings put forth a flexible and robust workflow for data-driven model discovery under uncertainty. All codes and data accompanying this manuscript are available at <https://bit.ly/34FOJMj>.

1. Introduction

In the era of big data, dynamical systems discovery has received a lot of attention, primarily due to the significant growth of accessible data across different scientific disciplines, including systems biology [1], bio-medical imaging [2], fluid dynamics [3], climate modeling [4], and physical chemistry [5]. Extracting a set of features that are

© The Authors. Published by the Royal Society under the terms of the Creative Commons Attribution License <http://creativecommons.org/licenses/by/4.0/>, which permits unrestricted use, provided the original author and source are credited.

interpretable and predictive is crucial for developing physical insight and gaining a better understanding of the natural phenomena under study [6]. Perhaps more importantly, this can enable the reliable forecasting of future states and subsequently lead to effective intervention strategies for design and control of complex systems [7–9].

Machine learning methods and data-driven modeling techniques have already proven their utility in solving high-dimensional problems in computer vision [10], natural language processing [11], etc. Due to their capability of extracting features from high dimensional and multi-fidelity noisy data [12,13], these methods are also gaining attraction in modeling and simulating physical and biological systems. The evolution of such systems can be typically characterized by differential equations, and several techniques have been developed to construct predictive algorithms that can synergistically combine data and mechanistic prior knowledge. Such scientific machine learning approaches are currently employed to distill dynamics from time-series data [14–21], infer the solution of differential equations [22–27], infer parameters, latent variables and unknown constitutive laws [3,28–31], as well as tackle forward and inverse problems in complex application domains including cardiovascular flow dynamics, [32], metamaterials [33], cardiac electrophysiology [34], etc.

Specific to systems identification, most recent data-driven approaches [14–17,21] heavily rely on the quality of the observations and are not designed to return predictions with quantified uncertainty. For instance, the sparse regression methods put forth in [17,18] can exhibit unstable behavior if the data is highly noisy and are not able to directly digest time-series data with irregular sampling frequency or missing values. On the other hand, recent approaches leveraging differentiable programming [14–16] can support irregularly sampled data, but are only designed to provide point-estimates for the discovered dynamics with no characterization of predictive uncertainty. Such lack of robustness and missing capabilities may limit the use of existing techniques to idealized settings and pose the need for a more flexible framework that can effectively accommodate noisy, sparse and irregularly sampled data to infer posterior distributions over plausible models, and subsequently yield robust future forecasts with quantified uncertainty.

In this work, we aim to address the aforementioned capability gaps by formulating a fully Bayesian framework for robust systems identification from imperfect time-series data. Our specific contributions can be summarized in the following points:

- We leverage recent developments in differentiable programming [14–16] to enable gradient back-propagation through numerical ODE solvers, and utilize this information to construct accelerated Hamiltonian Monte Carlo schemes for Bayesian inference.
- The proposed workflow is computationally efficient, end-to-end differentiable, and can directly accommodate sparse, noisy and irregularly sampled time-series data.
- Equipped with sparsity-promoting priors, we can recover interpretable and parsimonious representations for the latent dynamics, while entire posterior distributions over plausible models are obtained.
- This probabilistic formulation is key for safe-guarding against erroneous data, incomplete model parametrizations, as well as for producing reliable future forecasts with quantified uncertainty.
- We demonstrate enhanced capabilities and robustness against state-of-the-art methods for systems identification [17,18] across a range of benchmark problems.

Taken all together, our findings put forth a novel, flexible and robust workflow for data-driven model discovery under uncertainty that can potentially lead to improved algorithms for robust forecasting, control and model-based reinforcement learning of complex systems.

The rest of this paper is organized as follows. Section 2 presents the proposed method and its corresponding technical ingredients. In section (b), a review of Bayesian inference and Hamiltonian Monte Carlo sampling is given. In section (a), we provide details on how the proposed method can effectively propagate gradient information through ODE solvers leveraging

differential programming. In section (e), we discuss a simple but essential step for pre-processing the observed data in order to obtain robust training behavior. In section 3 the effectiveness of the proposed method is tested on a series of numerical studies, including nonlinear oscillators, predator-prey systems and a realistic example in systems biology. Finally, in section 4 we summarize our key findings, discuss the limitations of the proposed approach, and carve out directions for future investigation.

2. Methods

This section provides a comprehensive overview of the key ingredients that define our work, namely differential programming via Neural ordinary differential equations (NeuralODEs) [16] and Bayesian inference with Hamiltonian Monte Carlo sampling [35]. Our presentation is focused on describing how these techniques are be interfaced to obtain a novel, efficient, and robust workflow for parsimonious model discovery from imperfect time-series data.

(a) Differentiable programming

In their original work, Chen *et. al.* [16] introduced a general framework for propagating gradient information through classical numerical solvers for ordinary differential equations (ODEs) that blends classical adjoint methods [36] with modern developments in automatic differentiation [37]. To illustrate the main concepts, consider a general dynamical system of the form

$$\frac{d\mathbf{x}}{dt} = f(\mathbf{x}, t; \boldsymbol{\theta}), \quad (2.1)$$

where $\mathbf{x} \in \mathbb{R}^D$ denotes the state space of the D -dimensional dynamical system, and $\boldsymbol{\theta}$ is a vector unknown parameters that parametrizes the latent dynamics $f: \mathbb{R}^D \rightarrow \mathbb{R}^D$. A systems identification task is now summarized as follows. Given some observations $\mathbf{x}_i, i = 1, \dots, n$ evaluated at time instances $t_i, i = 1, \dots, n$, one would like to learn the $\boldsymbol{\theta}$ that best parametrizes the underlying dynamics. A typical approach for identifying these optimal parameters is to define a loss function g that measures the discrepancy between the observed data \mathbf{x}_{i+1} and the model's predictions $\hat{\mathbf{x}}_{i+1}$ for a given $\boldsymbol{\theta}$, i.e.,

$$\mathcal{J}(\boldsymbol{\theta}) = \sum_{i=1}^{n-1} g(\mathbf{x}_{i+1}, h_{\boldsymbol{\theta}}(\mathbf{x}(t_i))), \quad (2.2)$$

where $h_{\boldsymbol{\theta}}(\mathbf{x}(t_i)) = \hat{\mathbf{x}}_{i+1}$ is the predicted value under a given set of estimated model parameters $\boldsymbol{\theta}$ obtained by integrating the dynamical system with some ODE solver. $g(\cdot)$ could be any metric to evaluate the distance / evaluating the discrepancy between the true value and the predicted one (e.g., L_1 loss, L_2 loss [38] or KL-divergence [39], Wasserstein distance [40], etc). A sufficient way to minimize the loss function is through gradient descent [41,42], however appropriate methods need to be employed for effectively computing the gradient of the loss function with respect to the parameters, namely $\frac{\partial L}{\partial \boldsymbol{\theta}}$. This is done by defining the adjoint of the dynamical system as $\mathbf{a}(t) = \frac{\partial L}{\partial \mathbf{x}(t)}$. Then, the dynamical system describing the evolution of the adjoint can be derived as [16,36]

$$\frac{d\mathbf{a}(t)}{dt} = -\mathbf{a}(t)^T \frac{\partial f(\mathbf{x}(t), t, \boldsymbol{\theta})}{\partial \mathbf{x}}. \quad (2.3)$$

Note that the adjoint $\mathbf{a}(t)$ can be computed by an additional call to the chosen ODE solver, and the target derivative $\frac{\partial L}{\partial \boldsymbol{\theta}}$ can be then computed as

$$\frac{\partial L}{\partial \boldsymbol{\theta}} = - \int_{t_1}^{t_0} \mathbf{a}(t)^T \frac{\partial f(\mathbf{x}(t), t, \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} dt, \quad (2.4)$$

where $\frac{\partial f(\mathbf{x}(t), t, \boldsymbol{\theta})}{\partial \boldsymbol{\theta}}$ can be evaluated via automatic differentiation [16,37]. The main advantages of this approach can be summarized in the following points:

- The data does not need to be collected on a regular time grid.
- The time-step Δt_i between an observed data pair $\{\mathbf{x}(t_i), \mathbf{x}(t_i + \Delta t_i)\}$ can be relatively large. Within each Δt_i , a classical numerical scheme can be used to integrate equation 2.4, where $\Delta t_i = N_i dt$ is discretized in N_i equal spaced steps, with the step size dt being typically chosen according to the stability properties of the underlying ODE solver.
- As this setup only assumes dependency between individual input-output pairs $\{\mathbf{x}(t_i), \mathbf{x}(t_i + \Delta t_i)\}$, the observed time-series data does not need to be continuous and could be selected from different time intervals.

The specific choices of Δt_i , dt and N_i will be discussed for each of the different examples given in this work. In general, the choice of N is made based on the following trade-off between accuracy and computational complexity. To this end, small N may lead to a less accurate model, while large N would lead to a massive computational graph that can significantly slow down model training.

The unknown model parameters $\boldsymbol{\theta}$ can be estimated by minimizing appropriate loss function. Throughout this work, the following L_2 loss is employed

$$\mathcal{J}(\boldsymbol{\theta}) = \sum_{i=1}^n \|\mathbf{x}(t_i + \Delta t_i) - h_{\boldsymbol{\theta}}(\mathbf{x}(t_i))\|^2, \quad (2.5)$$

where $h_{\boldsymbol{\theta}}(\mathbf{x}(t_i))$ denotes the output of a numerical ODE solver. Throughout this work, we use the classical fourth order Runge-Kutta method [43], although more general choices can be employed [14]. The training data-set consists of pairs $\{(\mathbf{x}(t_1), \mathbf{x}(t_1 + \Delta t_1)), (\mathbf{x}(t_2), \mathbf{x}(t_2 + \Delta t_2)), \dots, (\mathbf{x}(t_n), \mathbf{x}(t_n + \Delta t_n))\}$. To simplify notation, let $\mathbf{X}(\mathbf{t})$ be defined as $\mathbf{X}(\mathbf{t}) = \{\mathbf{x}(t_1), \mathbf{x}(t_2), \dots, \mathbf{x}(t_n)\}$, such that $\mathbf{X}(\mathbf{t} + \Delta \mathbf{t}) = \{\mathbf{x}(t_1 + \Delta t_1), \mathbf{x}(t_2 + \Delta t_2), \dots, \mathbf{x}(t_n + \Delta t_n)\}$, then the training data-set is given by: $\mathcal{D} = \{\mathbf{X}(\mathbf{t}), \mathbf{X}(\mathbf{t} + \Delta \mathbf{t})\}$.

Notice that the aforementioned workflow is only capable of producing deterministic point estimates for the unknown parameters $\boldsymbol{\theta}$, typically corresponding to a local minimum of equation 2.5. In many practical cases it is desirable to obtain a distribution over plausible parameter configurations that can effectively characterize the uncertainty in the estimates due to noise and sparsity in the observed data, as well as potential misspecifications in the model parametrization. A framework for accounting for such uncertainties can be constructed by adopting a Bayesian approach via the use of effective sampling algorithms for approximating a posterior distribution over the unknown model parameters $p(\boldsymbol{\theta}|\mathcal{D})$, as discussed in the next section.

(b) Bayesian inference with Hamiltonian Monte Carlo

The Bayesian formalism provides a natural way to account for uncertainty, while also enabling the injection of prior information for the unknown model parameters $\boldsymbol{\theta}$ (e.g., sparsity), as well as for modeling sparse and noisy observations in the training data-set. Perhaps more importantly, it enables the complete statistical characterization for all inferred parameters in the model. The latter, is encapsulated in the posterior distribution which can be factorized as

$$p(\gamma, \lambda, \boldsymbol{\theta} | \mathbf{X}(\mathbf{t} + \Delta \mathbf{t}), \mathbf{X}(\mathbf{t})) \propto p(\mathbf{X}(\mathbf{t} + \Delta \mathbf{t}) | \mathbf{X}(\mathbf{t}), \boldsymbol{\theta}, \gamma) p(\boldsymbol{\theta} | \lambda) p(\gamma) p(\lambda), \quad (2.6)$$

where $p(\mathbf{X}(\mathbf{t} + \Delta \mathbf{t}) | \mathbf{X}(\mathbf{t}), \boldsymbol{\theta}, \gamma)$ is a likelihood function that measures the discrepancy between the observed data and the model's predictions for a given set of parameters $\boldsymbol{\theta}$, $p(\boldsymbol{\theta} | \lambda)$ is a prior distribution parametrized by a set of parameters λ that can help encode any domain knowledge about the unknown model parameters $\boldsymbol{\theta}$, and γ contains a set of parameters that aim to characterize the noise process that may be corrupting the observations. In this work, a hierarchical Bayesian approach corresponding to the following likelihood and priors is employed:

$$\begin{aligned}
p(\mathbf{X}(t + \Delta t) | \mathbf{X}(t), \boldsymbol{\theta}, \gamma) &= \prod_{i=1}^N \mathcal{N}(\mathbf{x}(t_i + \Delta t_i) | f(\mathbf{x}(t_i), t_i, \boldsymbol{\theta}), \gamma^{-1}), \\
p(\boldsymbol{\theta} | \lambda) &= \text{Laplace}(\boldsymbol{\theta} | 0, \lambda^{-1}), \\
p(\log \lambda) &= \text{Gam}(\log \lambda | \alpha_1, \beta_1), \\
p(\log \gamma) &= \text{Gam}(\log \gamma | \alpha_2, \beta_2).
\end{aligned} \tag{2.7}$$

The use of a Gaussian likelihood stems from assuming a simple isotropic Gaussian noise model with zero mean and precision γ . The Laplace prior over the unknown model parameters $\boldsymbol{\theta}$ [44] can promote sparsity in the inferred model representations and enable an effective reduction of the influence of any irrelevant parameters – a key feature for recovering interpretable and parsimonious representations [17]. The Gamma distribution is a common choice for the prior distributions over the unknown precision parameters λ and γ . For additional motivation and alternative choices, the interested reader is referred to [45–49]. Finally, the logarithm of the precision variables λ and γ is used to ensure that their estimated values remain positive during model training.

The posterior distribution defined in equation 2.7 is not analytically tractable in general due to the modeling assumptions on the likelihood and priors, as well as due to the presence of non-linearity in the latent dynamics of equation 2.1. Typically, sampling from this unnormalized distribution is difficult and computationally expensive, especially when the dimension of $\boldsymbol{\theta}$ is large. Hamiltonian Monte Carlo (HMC) [35] is a powerful tool to handle Bayesian inference tasks in high dimensions by utilizing gradient information to effectively generate approximate posterior samples. To illustrate the key ideas behind HMC sampling, let us denote $\boldsymbol{\Theta} = [\boldsymbol{\theta}, \log(\lambda), \log(\gamma)]$ as the vector including all the unknown parameters that need to be inferred from data. The starting point for building an HMC sampler is to define a Hamiltonian function $H = U(\boldsymbol{\Theta}) + V(\mathbf{v})$, where $U(\boldsymbol{\Theta})$ is the potential energy of the original system usually taken as the logarithm of the unnormalized distribution in equation 2.7, and $V(\mathbf{v})$ is the kinetic energy of the system introduced by the auxiliary velocity variables $\mathbf{v} := \frac{d\boldsymbol{\Theta}}{dt}$. The evolution of $\boldsymbol{\Theta}$ and \mathbf{v} can be expressed by taking gradients of the Hamiltonian as

$$\frac{d\mathbf{v}}{dt} = -\frac{\partial H}{\partial \boldsymbol{\Theta}}, \quad \frac{d\boldsymbol{\Theta}}{dt} = \frac{\partial H}{\partial \mathbf{v}}. \tag{2.8}$$

A Markov chain that converges to the stationary distribution of equation 2.6 can be simulated by integrating this dynamical system using an energy preserving leapfrog scheme [35]. Note here that computing the gradient $\frac{\partial H(t)}{\partial \boldsymbol{\Theta}}$ is not trivial as the evaluation of the likelihood function involves the call to an ODE solver. This difficulty can be directly addressed via the differentiable programming approach discussed in section (a) to effectively enable gradient back-propagation through the ODE solver. Note that all parameters in $\boldsymbol{\Theta}$ can be either updated simultaneously, or separately for $\boldsymbol{\theta}$ and $\{\lambda, \gamma\}$ using a Metropolis-within-Gibbs scheme [50,51].

(c) Learning dynamics with Bayesian differential programming

Here we distinguish between three different problem settings that cover a broad range of practical applications. The first class consists of problems in which the model form of the underlying latent dynamics is completely unknown. In this case, one can parametrize the unknown dynamical system using black-box function approximators such as deep neural networks [15,16,20], or aim to distill a more parsimonious and interpretable representation by constructing a comprehensive dictionary over all possible interactions and try to infer a predictive, yet minimal model form [17,18,21]. The second class of problems contains cases where a model form for the underlying dynamics is prescribed by domain knowledge, but a number of unknown parameters needs to be calibrated in order to accurately explain the observed data [31,52]. Finally, a third class of problems arises as a hybrid of the aforementioned settings, in which some parts of the model form

may be known from domain knowledge, but there exists additional functional terms that need to be inferred [14]. As detailed in the following sections, the proposed workflow can seamlessly accommodate all of the aforementioned cases in a unified fashion, while remaining robust with respect to incomplete model parametrizations, as well as imperfections in the observed data. To this end, a general framework can be constructed by parametrizing the unknown dynamics as

$$\frac{dx}{dt} = \underbrace{A\varphi(x)}_{\text{dictionary}} + \underbrace{f_w(x)}_{\text{black-box}}, \quad (2.9)$$

where $A \in \mathbb{R}^{D \times K}$ represents a matrix of $D \times K$ unknown coefficients, with K being the length of a dictionary $\varphi(x) \in \mathbb{R}^K$, that may or may not be constructed using domain knowledge. Specifically, $\varphi(x)$ represents the possible terms that may appear in the right hand side of the ordinary differential equation, which could encapsulate a known model form or, more generally, a prescribed dictionary of features (e.g., polynomials, Fourier modes, etc., and combinations thereof) [17,18,53]. On the other hand, $f_w(x)$ denotes a black-box function approximator (e.g., a neural network) parametrized by w that aims to account for any missing interactions that are not explicitly captured by $\varphi(x)$ [14].

The domain knowledge can play a crucial role in enhancing the efficiency of the resulting inference scheme as it can effectively reduce the size of the dictionary, and, consequently, the number of data required to train the model [23,24]. Such knowledge is also critical to constrain the space of admissible solutions such that key physical principles are faithfully captured by the inferred model (e.g., convergence to equilibrium limit cycles in chemical systems [54], conservation of mass and momentum in fluid dynamics [30], etc).

Although the use of dictionary learning (with or without specific domain knowledge) offers a flexible paradigm for recovering interpretable dynamic representations, it may not always be sufficient to explain the observed data, as important terms may be missing from the model parametrization. To address this shortcoming one can try to capture these missing interactions via the use of closure terms that often lack physical intuition, and hence can be represented by a black-box function approximator $f_w(x)$ with parameters w , such as a deep neural network [14]. Under this setup, the algorithmic framework outlined in sections (a) and (b) is employed to jointly perform probabilistic inference over plausible sets of model parameters $\theta := \{A, w\}$ that yield interpretable, parsimonious, and predictive representations, as well as the precision parameters λ and γ of the Bayesian hierarchical model in equation 2.7.

(d) Generating forecasts with quantified uncertainty

The goal of the HMC algorithm described in section (b) is to produce a faithful set of samples that concentrate in regions of high-probability in the posterior distribution $p(\theta, \lambda, \gamma | \mathcal{D})$. Approximating this distribution is central to the proposed workflow as it enables the generation of future forecasts $x^*(t)$ with quantified uncertainty via computing the predictive posterior distribution

$$p(x^*(t) | \mathcal{D}, x_0, t) = \int p(x^*(t) | \theta, x_0, t) p(\theta | \mathcal{D}) d\theta. \quad (2.10)$$

This predictive distribution provides a complete statistical characterization for the forecasted states by encapsulating epistemic uncertainty in the inferred dynamics, as well as accounting for the fact that the model was trained on a finite set of noisy observations. This allows the generation of plausible realizations of $x^*(t)$ by sampling from the predictive posterior distribution as

$$x^*(t) = h_\theta(x_0, t) + \epsilon, \quad \epsilon \sim \mathcal{N}(0, \gamma^{-1}), \quad \theta, \lambda, \gamma \sim p(\theta, \lambda, \gamma | \mathcal{D}) \quad (2.11)$$

where, θ , λ and γ are approximate samples from $p(\theta, \lambda, \gamma | \mathcal{D})$ computed during model training via HMC sampling, $h_\theta(x_0, t)$ denotes any numerical integrator that takes some initial condition x_0 and predicts the system's state at any time t , and ϵ accounts for the noise corrupting the

observations used during model training. Moreover, the maximum a-posteriori (MAP) estimate of the model parameters is given as follows:

$$\boldsymbol{\theta}_{\text{MAP}}, \lambda_{\text{MAP}}, \gamma_{\text{MAP}} = \arg \max_{\boldsymbol{\theta}, \lambda, \gamma} p(\boldsymbol{\theta}, \lambda, \gamma | \mathcal{D}), \quad (2.12)$$

and it is used to obtain a point estimate prediction of the predicted states $\hat{\mathbf{x}}_{\text{MAP}}(t)$ defined as following:

$$\mathbf{x}_{\text{MAP}}^*(t) = h_{\boldsymbol{\theta}_{\text{MAP}}}(\mathbf{x}_0, t). \quad (2.13)$$

Finally, it is straightforward to utilize the posterior samples of $\boldsymbol{\theta} \sim p(\boldsymbol{\theta} | \mathcal{D})$ to approximate the first- and second-order statistics of the predicted states $\mathbf{x}^*(t)$ for any given initial condition \mathbf{x}_0 as

$$\hat{\mu}_{\mathbf{x}^*}(\mathbf{x}_0, t) = \int h_{\boldsymbol{\theta}}(\mathbf{x}_0, t) p(\boldsymbol{\theta} | \mathcal{D}) d\boldsymbol{\theta} \approx \frac{1}{N_s} \sum_{i=1}^{N_s} h_{\boldsymbol{\theta}_i}(\mathbf{x}_0, t), \quad (2.14)$$

$$\hat{\sigma}_{\mathbf{x}^*}^2(\mathbf{x}_0, t) = \int [h_{\boldsymbol{\theta}}(\mathbf{x}_0, t) - \hat{\mu}_{\mathbf{x}^*}(\mathbf{x}_0, t)]^2 p(\boldsymbol{\theta} | \mathcal{D}) d\boldsymbol{\theta} \approx \frac{1}{N_s} \sum_{i=1}^{N_s} [h_{\boldsymbol{\theta}_i}(\mathbf{x}_0, t) - \hat{\mu}_{\mathbf{x}^*}(\mathbf{x}_0, t)]^2, \quad (2.15)$$

where N_s denotes the number of samples drawn from the Hamiltonian Markov Chain used to simulate the posterior, i.e., $\boldsymbol{\theta}_i \sim p(\boldsymbol{\theta} | \mathcal{D})$, $i = 1, \dots, N_s$. Note that higher-order moments are also readily computable in a similar manner.

(e) Model initialization and data pre-processing

To promote robustness and stability in the training of the proposed machine learning pipeline, users should be cognizant of several important aspects. First, although the proposed Bayesian approach can naturally safe-guard against over-fitting, it is important that a reasonable amount of training data is provided – relative to the complexity of the system – in order to mitigate any effects of prior misspecification. Second, the training data should be appropriately normalized in order to prevent gradient pathologies during back-propagation [55]. The specific utility of this step will be demonstrated in the numerical examples presented in this work, and is carried out using a standard normalization of the form

$$\tilde{\mathbf{x}} = \frac{\mathbf{x}}{\sigma_{\mathbf{x}}} \quad (2.16)$$

where $\sigma_{\mathbf{x}}$ is the dimension-wise standard deviation of the training data and the division is an element-wise operation. Notice that this modification directly implies that the assumed parametrization of the underlying dynamical system also needs to be normalized accordingly (see section 3 for a more detailed discussion). A third important remark here, is that the noise precision γ obtained from the model aims to reflect the noise level in the observed data. However, it may not be the true noise level because the initial condition of the ODE can also be noisy. This point will be further discussed in section 3.

Another important point relates to the initialization of the Hamiltonian Monte Carlo Markov Chain sampler. To this end, in order to mitigate poor mixing and convergence to local minima, a preconditioning step is considered to find a reasonable initial guess for the unknown variables $\boldsymbol{\theta}$ that parametrize the underlying latent dynamics. This step is typically carried out by minimizing the reconstruction loss of the training data using an L_1 regularization,

$$\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{n} \sum_{i=1}^n \|\mathbf{x}_i - \hat{\mathbf{x}}_i\|^2 + \beta \|\boldsymbol{\theta}\|, \quad (2.17)$$

using a small number ($\mathcal{O}(10^3)$) of stochastic gradient descent iterations. Notice that this essentially aims at obtaining a rough point estimate for $\boldsymbol{\theta}$, where the use of L_1 regularization stems from employing a sparsity-promoting Laplace prior. This preconditioning step is closely related to the SINDy algorithm of Brunton *et. al.* [17], however without the limitations of requiring training data with small noise amplitude and sampled on regular time grid with small a time step.

Moreover, the numerical experiments carried out indicate that tuning the hyper-parameter β has almost no effect on the obtained results for all the problems considered in this work, as this simply serves as an initialization step for the HMC sampler. Hence, in all examples considered, β is taken equal to 1. The minimization of equation 2.17 is carried out using stochastic Adam updates [56] with a learning rate of $\mathcal{O}(10^{-2})$.

Note that this preconditioning is not precisely equivalent to the MAP estimation of the posterior distribution over all model parameters Θ , since the initialization of the precision parameters λ and γ follows a different treatment. Specifically, for all the problems considered in section 3, the parameters α_i 's and β_i 's of the prior Gamma distributions are chosen to be 1. Moreover, the precision of the Gaussian noise distribution γ is initialized as follows. If the preconditioning precision is larger than $\exp(6)$, which means the training data appears to be nearly noise-free, the initial guess for γ is set to $\exp(6)$ to avoid numerical stagnancy of the HMC sampler. Otherwise, if the preconditioning precision is less than $\exp(6)$, then it is used as the initial guess for γ . This empirical initialization strategy has a positive effect in accelerating the convergence of the HMC sampler for all the examples considered in section 3. In all of the numerical examples, the Hamiltonian Monte Carlo step-size is taken as $\epsilon = 10^{-4}$, while the number number of leapfrog steps to integrate the Hamiltonian dynamics in equation 2.8 is fixed to $L = 10$. While this choice of ϵ is the safest choice, one can increase its value as long as not too many samples are rejected during model training. Alternatively, more sophisticated HMC samplers that allow for adaptively tuning the step-size can be employed [57]. Finally, in all examples the Markov Chains are simulated for 5,000 steps, while the last $N_s = 2,000$ samples produced by HMC are used to compute the response statistics (see equation 2.14).

3. Results

In this section, a comprehensive collection of numerical studies that aim to illustrate the key contributions of this work is presented and placed in context of the existing SINDy framework of Brunton *et. al.* [17], which is currently considered as a state-of-the-art method for dictionary learning of dynamical systems. Specifically, we expand on four benchmark problems that cover all possible cases discussed in section (c) in terms of parametrizing the latent dynamics using a dictionary, domain knowledge, black-box approximations, or a combination thereof. The algorithmic settings used across all cases follow the discussion provided in (e), unless otherwise noticed. All code and data presented in this section will be made publicly available at <https://github.com/PredictiveIntelligenceLab/BayesianDifferentiableProgramming>.

(a) Dictionary learning for a two-dimensional nonlinear oscillator

Let us start with a pedagogical example on dictionary learning for inferring the dynamics of a two-dimensional damped oscillator from scattered time-series data [20]. The exact system dynamics are given by

$$\begin{aligned} \frac{dx_1}{dt} &= \alpha x_1^3 + \beta x_2^3, \\ \frac{dx_2}{dt} &= \gamma x_1^3 + \delta x_2^3. \end{aligned} \quad (3.1)$$

The goal here is to recover this dynamical system directly from data using a dictionary parametrization containing polynomial terms with up to 3rd order interactions, taking the form

$$\begin{bmatrix} \frac{dx_1}{dt} \\ \frac{dx_2}{dt} \end{bmatrix} = A\varphi(\mathbf{x}) = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} & a_{15} & a_{16} & \textcolor{red}{a}_{17} & a_{18} & a_{19} & \textcolor{red}{a}_{110} \\ a_{21} & a_{22} & a_{23} & a_{24} & a_{25} & a_{26} & \textcolor{red}{a}_{27} & a_{28} & a_{29} & \textcolor{red}{a}_{210} \end{bmatrix} \varphi(\mathbf{x}) \quad (3.2)$$

where $\varphi(\mathbf{x}) = [1, x_1, x_2, x_1^2, x_1 x_2, x_2^2, x_1^3, x_1^2 x_2, x_1 x_2^2, x_2^3]^T$ and the a_{ij} 's are unknown scalar coefficients that will be estimated using the proposed Bayesian differential programming method. The model's active non-zero parameters are highlighted with red color for clarity. The goal

is to infer a posterior distribution for the parameters $\theta := \{A\}$, while it is obvious that the coefficient matrix A and dictionary $\varphi(\mathbf{x})$ will increase in size with the order of the polynomial features used to parametrize the system. In all presented experiments, a set of training data is generated by simulating the exact dynamics of equation 3.1 in the time interval $t \in [0, 20]$ with the initial condition $(x_1, x_2) = (2, 0)$ and with $\alpha = -0.1$, $\beta = 2.0$, $\gamma = -2.0$, and $\delta = -0.1$. In what follows, the performance of the proposed algorithms is investigated with respect to the temporal resolution and the noise level in the observed data. Moreover, the analysis is provided with a comprehensive comparison with the SINDy algorithm of Brunton *et. al.* [17].

(i) Effects of sparsity in the training data

To examine the performance of the proposed methods with respect to sparsity in the training data, a data-set is generated by integrating the exact dynamics with a relatively large $dt = 0.0677$, such that there are only $n = 300$ training data pairs. In order to establish a comparison against the SINDy algorithm [17], the training data is assumed noise-free and sampled on a regular temporal grid, as required by the SINDy setup [17].

As seen in figures 1 and 2, the SINDy algorithm fails to identify the underlying system, while the proposed approach remains robust thanks to the Bayesian formulation outlined in section (b) and which enables a faithful statistical characterization of the latent system dynamics even under sparse observations. Indeed, the proposed method does not only have the capability of accurately identifying the parameters even for relatively large dt (see table 1), but also provides reasonable uncertainty estimates for the extrapolated long-term forecasts. In contrast, the SINDy algorithm gives inaccurate estimations for the dictionary parameters (see table 2), consequently leading to large errors in the forecasted system states. As shown in figures 1(b),(d), the estimated trajectories clearly deviate from the exact solution since its oscillatory frequency is considerably higher than the exact one, and it is not capturing the decay of the oscillations' peak. Moreover, SINDy's results highly depend on the choice of the sequential least-squares threshold parameter value, and slightly different values of this hyper-parameter may give significantly different results.

Table 1: *Dictionary learning for a two-dimensional nonlinear oscillator:* MAP estimation of the inferred model parameters using low-resolution training data ($dt = 0.0677$).

a_{11}	a_{12}	a_{13}	a_{14}	a_{15}	a_{16}	a_{17}	a_{18}	a_{19}	a_{110}
-0.0052	0.0057	-0.0095	0.0094	0.0116	0.0057	-0.097	-0.0093	-0.054	2.092
a_{21}	a_{22}	a_{23}	a_{24}	a_{25}	a_{26}	a_{27}	a_{28}	a_{29}	a_{210}
0.0048	0.0195	-0.0072	-0.021	0.0024	-0.0075	-2.08	0.0280	-0.014	-0.099

Table 2: *Dictionary learning for a two-dimensional nonlinear oscillator:* Point estimates for the dictionary coefficients obtained by the SINDy algorithm [17] using low-resolution training data ($dt = 0.0677$).

a_{11}	a_{12}	a_{13}	a_{14}	a_{15}	a_{16}	a_{17}	a_{18}	a_{19}	a_{110}
0	0	0	0	0	0	0	0	0	2.10
a_{21}	a_{22}	a_{23}	a_{24}	a_{25}	a_{26}	a_{27}	a_{28}	a_{29}	a_{210}
0	0	0	0	0	0	-1.87	0	0	0

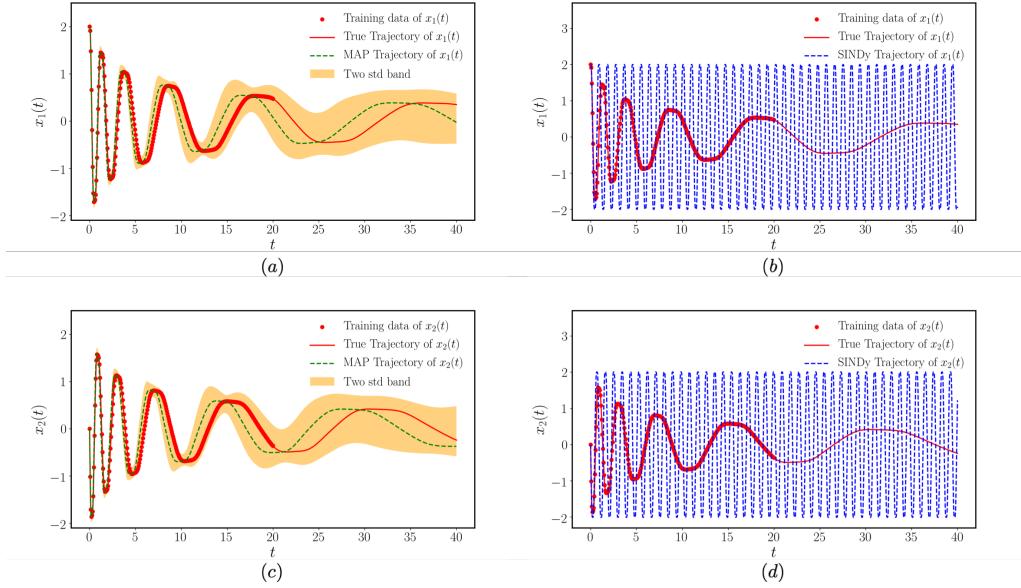


Figure 1: Two-dimensional damped oscillator with low-resolution training data: (a) Learned dynamics versus the true dynamics and the training data for $x_1(t)$. (b) SINDy's prediction for $x_1(t)$ versus the true dynamics and the training data. (c) Learned dynamics versus the true dynamics and the training data for $x_2(t)$. (d) SINDy's prediction for $x_2(t)$ versus the true dynamics and the training data.

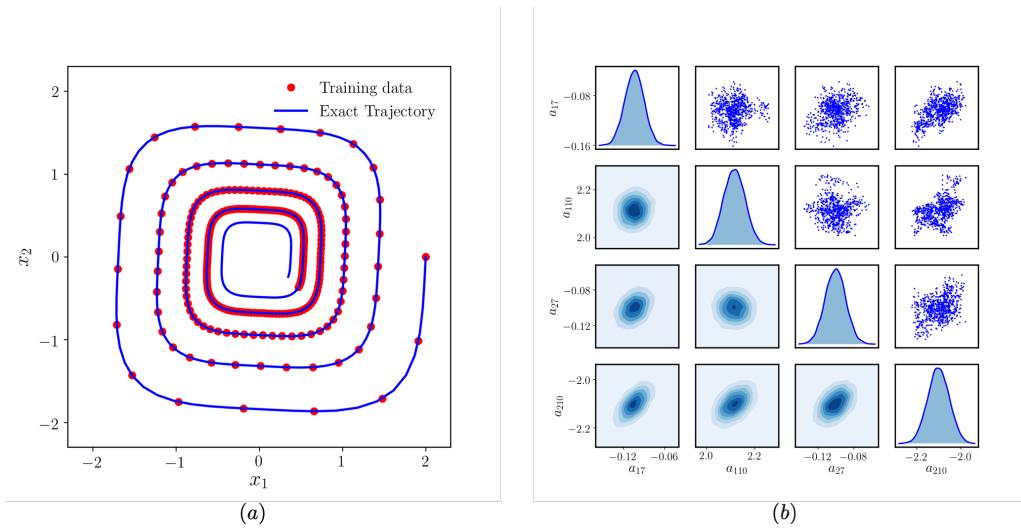


Figure 2: Two-dimensional damped oscillator with low-resolution training data: (a) Phase plot of the training data and the true trajectory. (b) Posterior distribution of the inferred active model parameters.

(ii) Effects of noise in the training data

In this section, the sensitivity of the proposed methods with respect to the presence of noise in the training data is investigated. To this end, a training data-set is generated with $n = 1,000$ equi-spaced data-pairs in $t \in [0, 20]$ by simulating the exact dynamics of equation 3.1 with $dt = 0.02$, and the observations are deliberately corrupted with uncorrelated Gaussian noise of the form $\mathcal{N}(0, 0.02^2)$ (see figure 4(a)).

Figure 3 summarizes the predictions of the proposed Bayesian framework in comparison to the SINDy algorithm of Brunton *et. al.* [17]. Moreover, the inferred dictionary parameters are provided for both methods in tables 3 and 4, respectively. Notice that, although the predicted trajectories of SINDy are quite close to the true trajectories, the identified parameters are quite different from the true model form. Even though SINDy employs a total variation diminishing (TVD) regularization to safe-guard against small noise corruptions in the training data [17], it is still prone to providing inaccurate results in cases where the training data is imperfect. This limitation is addressed in the proposed framework by explicitly accounting for the effects of noise in the training data using the hierarchical Bayesian model in equation 2.6. This source of uncertainty is also effectively propagated through the system's dynamics to yield a sensible characterization of uncertainty in the predicted forecasts (see figure 3(a)), via the inferred posterior distribution over the model parameters (see figure 4(b)). Moreover, the resulting MAP estimates for the model parameters exhibit excellent agreement with the ground truth, as reported in table 3. This result illustrates the robust performance of the proposed framework in identifying interpretable and parsimonious system representations, even in the presence of noise in the training data.

Table 3: *Dictionary learning for a two-dimensional nonlinear oscillator:* MAP estimation of the inferred model parameters using noisy training data.

a_{11}	a_{12}	a_{13}	a_{14}	a_{15}	a_{16}	a_{17}	a_{18}	a_{19}	a_{110}
0.0042	-0.0059	-0.0045	-0.0072	-0.0078	-0.0071	-0.10	0.047	0.014	2.0
a_{21}	a_{22}	a_{23}	a_{24}	a_{25}	a_{26}	a_{27}	a_{28}	a_{29}	a_{210}
-0.0029	-0.010	-0.0073	0.0017	0.017	0.0064	-2.0	0.015	-0.049	-0.108

Table 4: *Dictionary learning for a two-dimensional nonlinear oscillator:* estimation of the SINDy's parameters using noisy training data.

a_{11}	a_{12}	a_{13}	a_{14}	a_{15}	a_{16}	a_{17}	a_{18}	a_{19}	a_{110}
0	0	0	0	0	0	-0.139	0	0	2.0
a_{21}	a_{22}	a_{23}	a_{24}	a_{25}	a_{26}	a_{27}	a_{28}	a_{29}	a_{210}
0	0	0	0	0	0	-1.97	0	-0.188	0

(b) Parameter inference in a predator-prey system with irregularly sampled observations

This case study is designed to illustrate the capability of the proposed framework to accommodate noisy and irregularly sampled time-series data; a common practical setting that cannot be effectively addressed by SINDy and other popular data-driven systems identification methods [17,18,20,21,58]. To this end, a classical prey-predator system described by the Lotka-Volterra

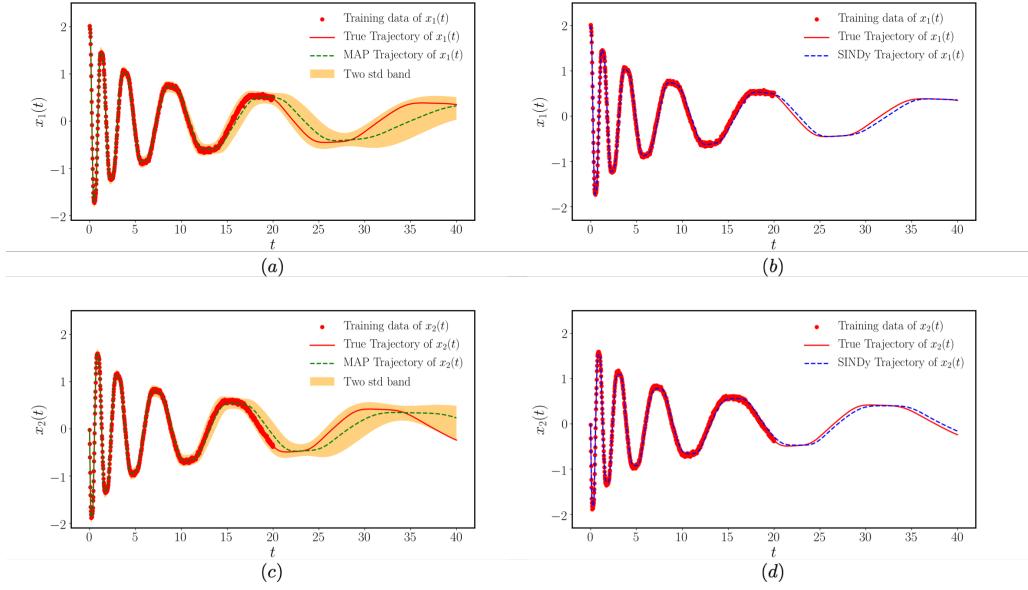


Figure 3: Two-dimensional damped oscillator with noisy training data: (a) Learned dynamics versus the true dynamics and the training data for $x_1(t)$. (b) SINDy's prediction for $x_1(t)$ versus the true dynamics and the training data. (c) Learned dynamics versus the true dynamics and the training data for $x_2(t)$. (d) SINDy's prediction for $x_2(t)$ versus the true dynamics and the training data.

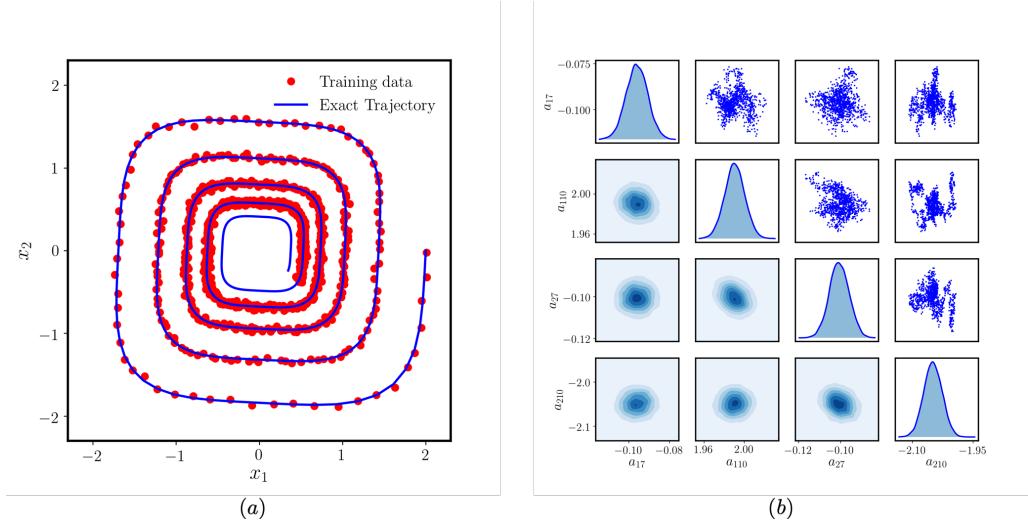


Figure 4: Two-dimensional damped oscillator with noisy training data: (a) Phase plot of the training data and the true trajectory. (b) Posterior distribution of the inferred active model parameters.

equations is considered

$$\begin{aligned} \frac{dx_1}{dt} &= \alpha x_1 - \beta x_1 x_2 , \\ \frac{dx_2}{dt} &= \delta x_1 x_2 - \gamma x_2 , \end{aligned} \quad (3.3)$$

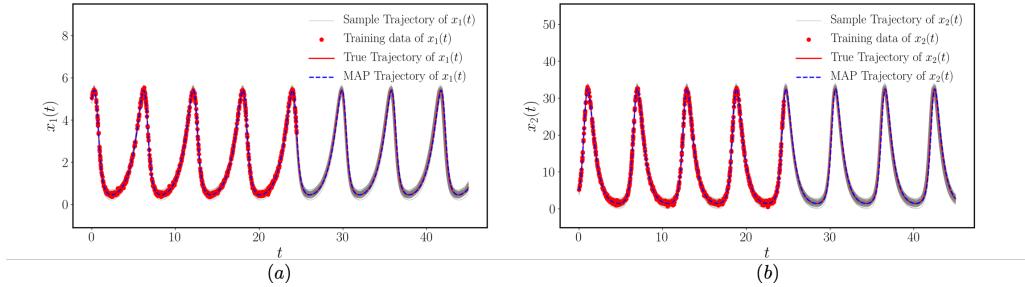


Figure 5: *Parameter inference in a predator-prey system from irregularly sampled, noisy data:* (a) Learned dynamics versus the true dynamics and the training data for x_1 . (b) Learned dynamics versus the true dynamics and the training data for x_2 .

which is known to exhibit a stable limit cycle behavior for $\alpha = 1.0$, $\beta = -0.1$, $\gamma = -1.5$, and $\delta = 0.75$. Without loss of generality, a dictionary that precisely contains the active terms of the system is considered. However, the $n = 1000$ training data-pairs will be irregularly sampled in the interval $t \in [0, 25]$ by randomly sub-sampling an exact trajectory of system 3.3 starting from an initial condition set to $(x_1, x_2) = (5, 5)$. Moreover, the training data is perturbed by 3% white noise proportional to its standard deviation. Given this irregular and noisy training data, the goal is to demonstrate the performance of the proposed Bayesian framework in identifying the unknown model parameters $\theta := \{\alpha, \beta, \gamma, \delta\}$ with quantified uncertainty, as well as in producing sensible forecasts of extrapolated future states.

The results of this experiment are summarized in figures 5 and 6. It is evident that the proposed Bayesian differential programming approach (i) is able to provide an accurate estimation for the unknown model parameters, (ii) yields a MAP estimator with a predicted trajectory that closely matches the exact system's dynamics, (iii) returns a posterior distribution over plausible models that captures both the epistemic uncertainty of the assumed parametrization and the uncertainty induced by training on a finite amount of noisy training data, and (iv) propagates this uncertainty through the system's dynamics to characterize variability in the predicted future states.

Another interesting observation here is that the Hamiltonian Monte Carlo sampler is very efficient in identifying the importance/sensitivity of each inferred parameter in the model. For instance, less important parameters have the highest uncertainty, as observed in the posterior density plots shown in figures 6(b). Specifically, notice how the posterior distribution of α and γ has a considerably larger standard deviation than the other parameters, implying that the evolution of this dynamical system is less sensitive with respect to these parameters.

(c) Safe-guarding against model inadequacy: a damped pendulum case study

The purpose of this example is to demonstrate the effects of a misspecified model parametrization, and to highlight how the proposed Bayesian framework can help detect such problematic cases and safe-guard against them. Such cases may arise when domain knowledge is insufficient to guide the selection of a parsimonious model form, as well as when important interaction terms may be missing from a dictionary representation. To illustrate the main ideas, a simple damped pendulum system described by the following equation is considered:

$$\begin{aligned} \frac{dx_1}{dt} &= \gamma x_2, \\ \frac{dx_2}{dt} &= -\alpha x_2 - \beta \sin(x_1), \end{aligned} \tag{3.4}$$

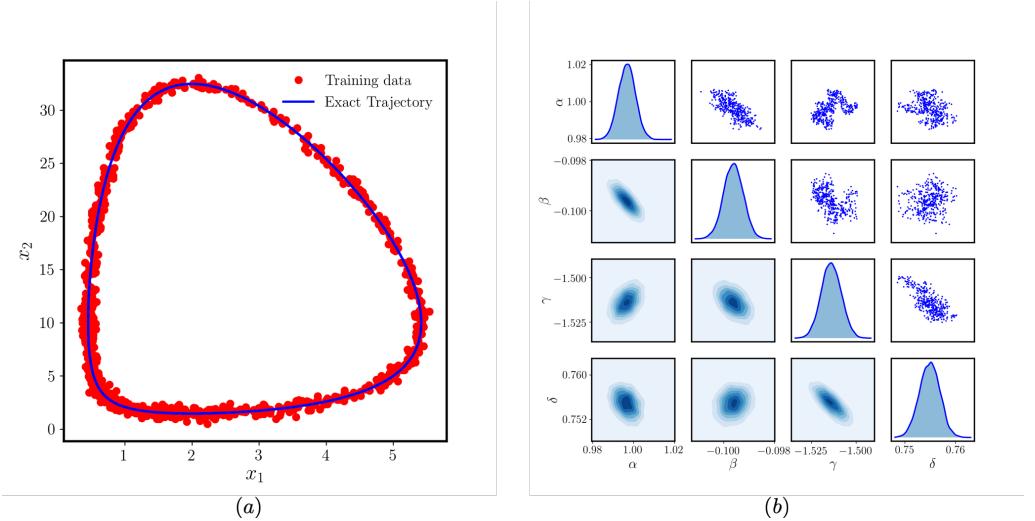


Figure 6: *Parameter inference in a predator-prey system from irregularly sampled, noisy data:* (a) Phase plot of the training data and the true trajectory. (b) Posterior distribution of the inferred model parameters.

with $\gamma = 1$, $\alpha = 0.2$ and $\beta = 8.91$. A set of sparse and irregularly sampled training data-pairs can be generated by randomly sub-sampling a simulated trajectory of the exact dynamics in $t \in [0, 20]$, starting from an initial condition $(x_1, x_2) = (-1.193, -3.876)$. This imperfect data-set can be used to recover an interpretable model representation via dictionary learning, albeit here we deliberately choose to use an incomplete dictionary containing polynomial terms only up to 1st order, i.e.,

$$\begin{bmatrix} \frac{dx_1}{dt} \\ \frac{dx_2}{dt} \end{bmatrix} = A\varphi(\boldsymbol{x}) = \begin{bmatrix} a_{11} & a_{12} & \color{red}{a_{13}} \\ a_{21} & \color{blue}{a_{22}} & \color{red}{a_{23}} \end{bmatrix} \begin{bmatrix} 1 \\ x_1 \\ x_2 \end{bmatrix}, \quad (3.5)$$

where the unknown model parameters are $\boldsymbol{\theta} := \{a_{11}, a_{12}, a_{13}, a_{21}, a_{22}, a_{23}\}$, with the active terms being marked with red color for clarity. Moreover, a blue marker is used to highlight a mismatched term in the incomplete dictionary, hinting its erroneous capacity to approximate the true $\sin(x_1)$ term using just x_1 as a feature. It is evident that such a dictionary choice cannot faithfully capture the exact physics of the problem, and is hence destined to yield inaccurate predictions. Nevertheless, here we argue that this discrepancy between the true form and the assumed incomplete parametrization of the dynamical system can be effectively detected by the proposed Bayesian workflow via inspecting the inferred posterior distribution over all parameters $p(\boldsymbol{\theta}, \lambda, \gamma | \mathcal{D})$, which is expected to exhibit high entropy in presence of a misspecified model parametrization and imperfect training data.

The results of this experiment are summarized in figures 7 and 8. In particular, figure 7 shows the scattered training data, the exact simulated trajectory, the predicted trajectory corresponding to the MAP estimate of the inferred model parameters, as well as a two standard deviations band around a set of plausible forecasted states. As expected, the use of a misspecified dictionary leads to an inferred model that has difficulty in fitting the observed data and yields predictions with high variability. This model inadequacy is also clearly captured in the posterior distribution over the inferred parameters shown in figure 8. Indeed, the inferred density for the β parameter exhibits very high variance, as the β coefficient crucially corresponds to the misspecified term in the dictionary. This is a direct indication that the inferred model is inadequate to capture the observed reality, leading to forecasts with inevitably large error-bars. This innate ability of the proposed framework to detect model misspecification via a rigorous quantification of predictive

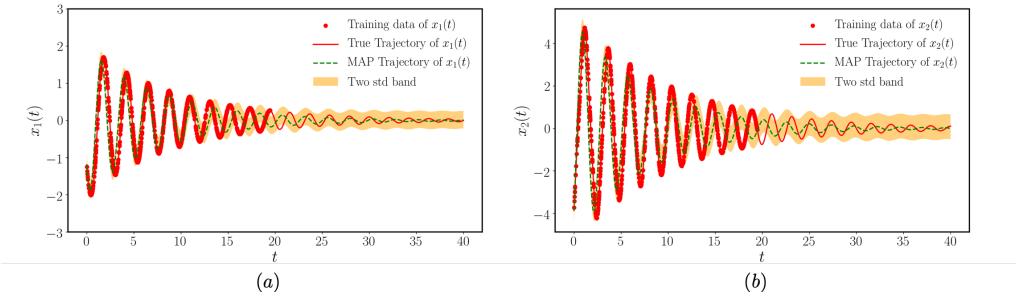


Figure 7: *Damped pendulum with irregularly sampled data:* (a) Learned dynamics versus the true dynamics and the training data for $x_1(t)$. (b) Learned dynamics versus the true dynamics and the training data for $x_2(t)$.

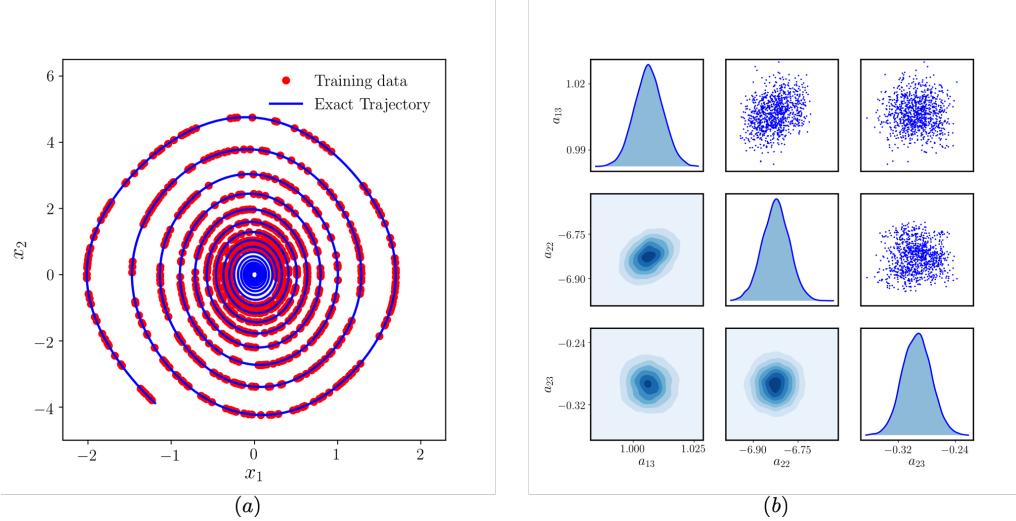


Figure 8: *Damped pendulum with irregularly sampled data:* (a) Phase plot of the training data and the true trajectory. (b) Posterior distribution of the model's parameters.

uncertainty can prove crucial in risk-sensitive applications, and provides an important capability that is currently missing from the recently active literature on data-driven model discovery [14,17,18,20,21,53].

A natural question to now ask is: can we still recover an accurate predictive model even if the true dynamics can only be partially captured by the assumed dictionary representation? To tackle this question we turn our attention to the hybrid learning setting discussed in section (c), in which some parts of the model can be captured by sparsely selecting interpretable terms from a dictionary, while other missing parts or closure terms can be accounted for via a black-box function approximator. To this end, the damped pendulum case study is revisited, and the dictionary learning parametrization is endowed with the ability to approximate the missing $\sin(x_1)$ term via a deep neural network as

$$\begin{bmatrix} \frac{dx_1}{dt} \\ \frac{dx_2}{dt} \end{bmatrix} = A\varphi(\mathbf{x}) + f_{\mathbf{w}}(\mathbf{x}) = \begin{bmatrix} a_{11} + a_{12}x_1 + \textcolor{red}{a}_{13}x_2 + a_{14}x_1^2 + a_{15}x_1x_2 + a_{16}x_2^2 \\ \textcolor{red}{a}_{21}x_2 + a_{22}x_1x_2 + a_{23}x_2^2 + f_{\mathbf{w}}(x_1), \end{bmatrix} \quad (3.6)$$

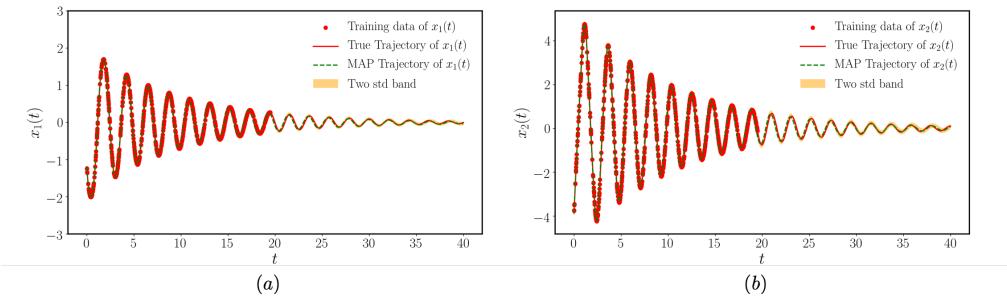


Figure 9: *Safe-guarding against model inadequacy in damped pendulum representation:* (a) Learned dynamics versus the true dynamics and the training data for $x_1(t)$. (b) Learned dynamics versus the true dynamics and the training data for $x_2(t)$. A deep neural network is used to approximate any interactions missing from an incomplete dictionary representation.

where A is a matrix of unknown coefficients corresponding to a dictionary $\varphi(\mathbf{x})$ containing polynomial interactions up to 2nd order, and $f_{\mathbf{w}}(x_1)$ is a fully-connected deep neural network with 2 hidden layers of dimension 20, a hyperbolic tangent activation, and a set of unknown weight and bias parameters denoted by \mathbf{w} . Notice that the neural network admits only x_1 as an input to make sure that the resulting parametrized dynamical system has a unique solution. Under this setup, the proposed Bayesian framework can be employed to jointly infer the dictionary and the neural network parameters that define the model representation, namely $\boldsymbol{\theta} := \{a_{11}, a_{12}, a_{13}, a_{14}, a_{15}, a_{16}, a_{21}, a_{22}, a_{23}, \mathbf{w}\}$. This defines a 490-dimensional inference problem.

Figure 9 shows the scattered training data, the exact simulated trajectory, the predicted trajectory corresponding to the MAP estimate of the inferred model parameters, as well as a two standard deviations band around a set of plausible forecasted states. It is evident that the revised hybrid model formulation can now correctly identify the true underlying dynamics, leading to an accurate predicted MAP trajectory, while the predicted uncertainty effectively diminishes and concentrates around the ground truth. Moreover, the inferred coefficients of the irrelevant terms are very close to zero, while the active terms are all properly identified. This is also true for the neural network approximation of the missing closure term $-\beta \sin(x_1)$, as depicted in figure 10 which also includes uncertainty estimates over the neural network outputs. This simple example illustrates the great flexibility of the proposed Bayesian framework in seamlessly distilling parsimonious and interpretable models from imperfect data via physics-informed dictionary learning, as well harnessing the power of black-box representations for approximating missing closure terms.

(d) Bayesian calibration of a Yeast Glycolysis model

In this final example, the performance of the proposed algorithms applied to a realistic problem in systems biology is investigated. To this end, a yeast glycolysis process is considered and described

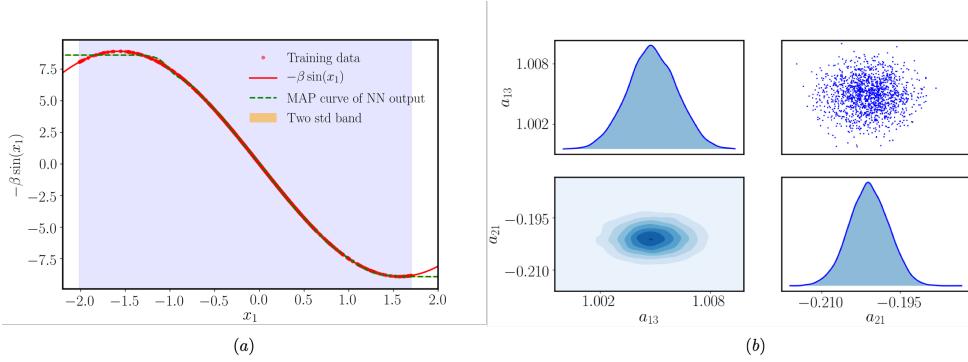


Figure 10: *Safe-guarding against model inadequacy in damped pendulum representation:* (a) Deep neural network approximation to the missing dictionary term $-\beta \sin(x_1)$. The shade region indicated the range of data seen during model training. (b) Posterior distribution of the inferred active model parameters.

by a 7-dimensional dynamical system [1,52] as:

$$\begin{aligned} \frac{dS_1}{dt} &= J_0 - k_1 S_1 A_3 [1 + (\frac{A_3}{K_I})^q]^{-1}, \\ \frac{dS_2}{dt} &= 2k_1 S_1 A_3 [1 + (\frac{A_3}{K_I})^q]^{-1} - k_2 S_2 N_1 - k_6 S_2 N_2, \\ \frac{dS_3}{dt} &= k_2 S_2 N_1 - k_3 S_3 A_2, \\ \frac{dS_4}{dt} &= k_3 S_3 A_2 - k_4 S_4 N_2, -\kappa(S_4 - S_4^{ex}), \\ \frac{dN_2}{dt} &= k_2 S_2 N_1 - k_4 S_4 N_2, -k_6 S_2 N_2, \\ \frac{dA_3}{dt} &= -2k_1 S_1 A_3 [1 + (\frac{A_3}{K_I})^q]^{-1} + 2k_3 S_3 A_2 - k_5 A_3, \\ \frac{dS_4^{ex}}{dt} &= \phi\kappa(S_4 - S_4^{ex}) - kS_4^{ex}, \end{aligned} \quad (3.7)$$

where $N_1 + N_2 = N$ and $A_2 + A_3 = A$. This model form is assumed to be known from existing domain knowledge, and the goal is (i) to compute the posterior distribution over the unknown model parameters $\theta := \{J_0, k_1, k_2, k_3, k_4, k_5, k_6, \kappa, q, K_I, \phi, N, A\}$ from a small set of noisy and irregularly sampled observations, and (ii) to test the ability of the inferred model to accurately generalize from different initial conditions that were not observed during model training.

A training data-set is generated by randomly sub-sampling $n = 1,000$ irregularly distributed observations from a single simulated trajectory of the system from an initial condition: $(S_1, S_2, S_3, S_4, N_2, A_3, S_4^{ex}) = (0.5, 1.9, 0.18, 0.15, 0.16, 0.1, 0.064)$ in the time interval $t \in [0, 5]$, assuming a ground truth set of parameters obtained from the experimental data provided in [1]: $J_0 = 2.5\text{mM/min}$, $k_1 = 100.0\text{mM/min}$, $k_2 = 6.0\text{mM/min}$, $k_3 = 16.0\text{mM/min}$, $k_4 = 100.0\text{mM/min}$, $k_5 = 1.28/\text{min}$, $k_6 = 12.0\text{mM/min}$, $\kappa = 1.8/\text{min}$, $q = 4.0$, $K_I = 0.52\text{mM}$, $N = 1.0\text{mM}$, $A = 4.0\text{mM}$ and $\phi = 0.1$. Moreover, the training data is perturbed by a 2% white noise proportional to its standard deviation.

Table 5 summarizes the inferred MAP estimators for the unknown model parameters. Based on those results, all inferred parameters closely agree with the ground truth values used to generate the training data as reported in [1]. Uncertainty estimates for the inferred parameters can also be deducted from the computed posterior distribution $p(\theta|\mathcal{D})$, as presented in the box plots of figure 11 where the minimum, maximum, median, first quantile and third quantile obtained from the

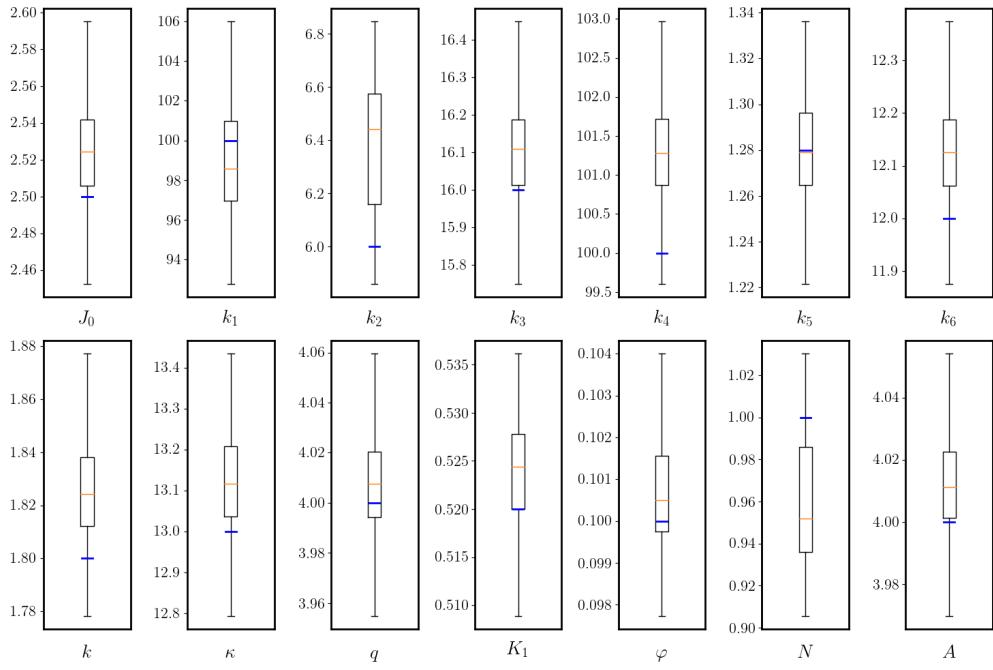


Figure 11: *Yeast Glycolysis dynamics*: Uncertainty estimation of the inferred model parameters obtained using the proposed Bayesian differential programming method. Estimates for the minimum, maximum, median, first quantile and third quantile are provided, while the true parameter values are highlighted in blue.

HMC simulations for each parameter are presented. Finally, notice that all true values fall between the predicted quantiles, while the MAP estimators of all the parameters have considerably small relative errors compared with the true parameter values, as shown in table 5.

Table 5: *Yeast Glycolysis dynamics*: MAP estimation of the inferred model parameters using noisy training data.

J_0	k_1	k_2	k_3	k_4	k_5	k_6	k	κ	q	K_1	ϕ	N	A
2.52	98.92	6.43	16.12	101.24	1.28	12.13	1.82	13.12	4.01	0.52	0.01	0.95	4.01

Finally, to illustrate the generalization capability of the inferred model with respect to different initial conditions than those used during training, the quality of the predicted states is assessed considering a random initial condition of $[0.428, 1.42, 0.11, 0.296, 0.252, 0.830, 0.064]$, that has not been observed during model training. The close agreement with the reference solution indicates that the inferred model is well capable of generalizing both in terms of handling different initial conditions, as well as extrapolating to reliable future forecasts with quantified uncertainty.

4. Conclusions

We have presented a novel machine learning framework for robust systems identification under uncertainty. The proposed framework leverages state-of-the-art differential programming techniques in combination with gradient-enhanced sampling schemes for scalable Bayesian inference of high-dimensional posterior distributions, to infer interpretable and parsimonious

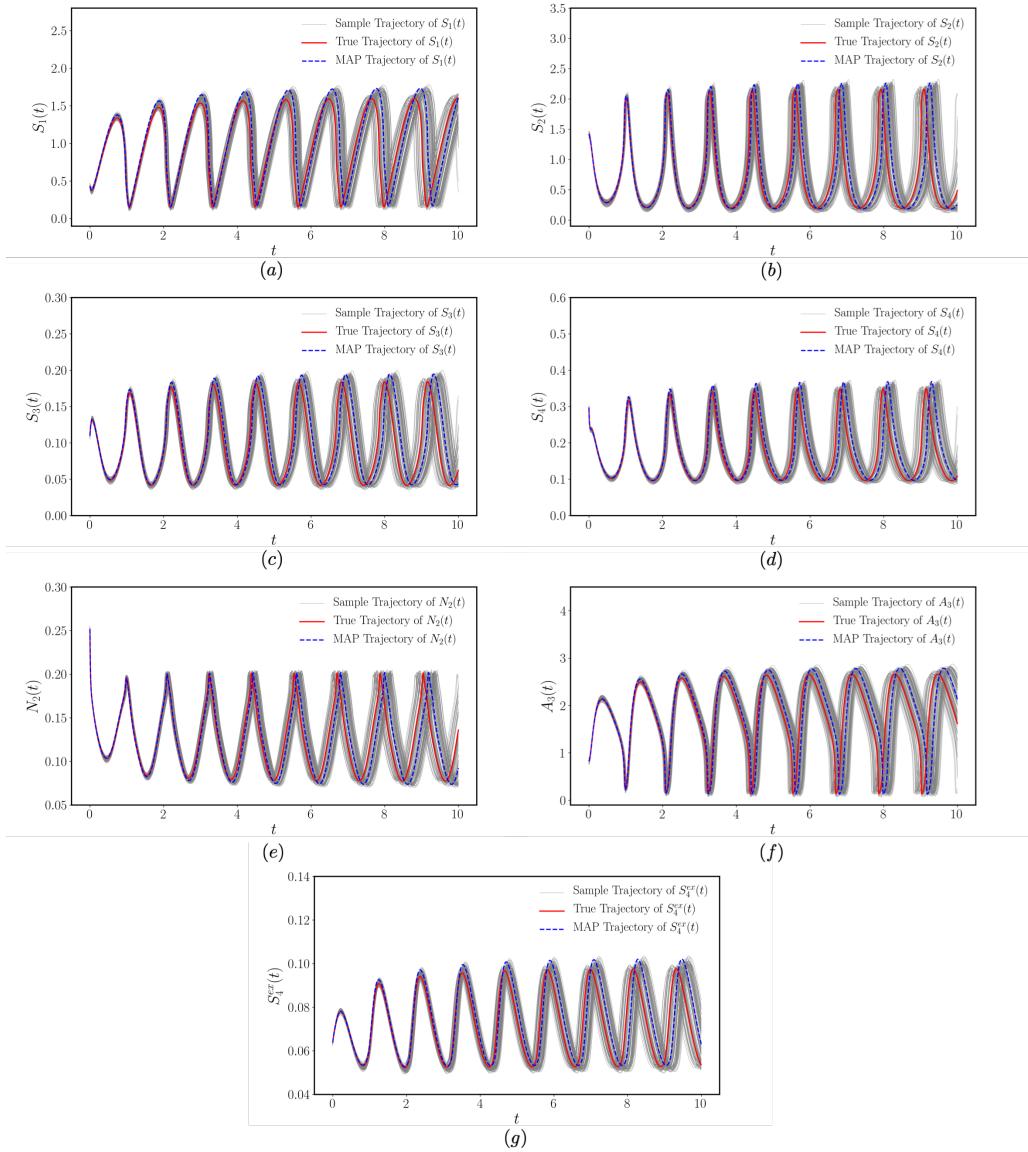


Figure 12: *Learning Yeast Glycolysis dynamics from noisy data:* Future forecasts with quantified uncertainty from a previously unseen initial condition (i.e. an initial condition that was not used during model training).

representations of complex dynamical systems from *imperfect* (e.g. sparse, noisy, irregularly sampled) data. The developed methods are general as they can seamlessly combine dictionary learning, domain knowledge and black-box approximations, all in a computationally efficient workflow with end-to-end uncertainty quantification. The effectiveness of the proposed techniques has been systematically investigated and compared to state-of-the-art approaches across a collection of prototype problems.

Although the proposed Bayesian differential programming framework provides great flexibility to infer a distribution over plausible parsimonious representations of a dynamical system, a number of technical issues need to be further investigated. The first relates to devising more effective initialization procedures for Markov Chain Monte Carlo sampling. Here we have

partially addressed this via the MAP preconditioning algorithm discussed in section (e), however during the preconditioning process, since the form of the dynamical system is unknown, the intermediate estimations of the parameters may cause the system to become stiff. Moreover, for cases where only very sparse observations are available, the model needs to be integrated with a large time-step dt and stiffness of the system can lead to numerical instabilities during model training. A possible enhancement in this direction is to use more general stiffly stable ODE solvers as discussed in [14,15] or more sophisticated time-step annealing strategies [27]. Another potential future work could be identifying the uncertainty in model's parameters with partial observations, such that some variables of the system are not accessible. Such task would involve physics-informed regularization on the unknown latent dynamics of the system. Approaches used in [52] could be helpful for solving this problem. A third open question is how to adapt the proposed method to stochastic dynamical systems where the dynamics itself may be driving by a stochastic process. Approaches mentioned in [59] could be useful. Finally, the proposed Bayesian differential programming framework can be extended to parameter identification for partial differential equations (PDEs). The latter generally translates into a high dimensional dynamical systems after discretization. The learning task in this context could be carried out not only for the PDEs' parameters, but also for the discretization scheme [14].

Data Accessibility. All code and data accompanying this manuscript is available at <https://github.com/PredictiveIntelligenceLab/BayesianDifferentiableProgramming>.

Authors' Contributions. P.P. and Y.Y. conceived the methods, Y.Y. implemented the methods, Y.Y. and M.A.B. performed the simulations. Y.Y., M.A.B. and P.P. wrote the manuscript.

Competing Interests. The authors have no competing interests to declare.

Funding. This work received support from the US Department of Energy under the Advanced Scientific Computing Research program (grant DE-SC0019116) and the Defense Advanced Research Projects Agency under the Physics of Artificial Intelligence program (grant HR00111890034).

References

- Ruoff P, Christensen MK, Wolf J, Heinrich R. 2003 Temperature dependency and temperature compensation in a model of yeast glycolytic oscillations. *Biophysical chemistry* **106**, 179–192.
- Kak AC, Slaney M, Wang G. 2002 Principles of computerized tomographic imaging. *Medical Physics* **29**, 107–107.
- Raissi M, Yazdani A, Karniadakis GE. 2020 Hidden fluid mechanics: Learning velocity and pressure fields from flow visualizations. *Science* **367**, 1026–1030.
- Palmer TN. 1999 A nonlinear dynamical perspective on climate prediction. *Journal of Climate* **12**, 575–591.
- Feinberg M, Horn FJ. 1974 Dynamics of open chemical systems and the algebraic structure of the underlying reaction network. *Chemical Engineering Science* **29**, 775–787.
- Haller G. 2002 Lagrangian coherent structures from approximate velocity data. *Physics of fluids* **14**, 1851–1861.
- Tantet A, Lucarini V, Lunkeit F, Dijkstra HA. 2018 Crisis of the chaotic attractor of a climate model: a transfer operator approach. *Nonlinearity* **31**, 2221.
- Bemporad A, Morari M. 1999 Control of systems integrating logic, dynamics, and constraints. *Automatica* **35**, 407–427.
- Lu F, Zhong M, Tang S, Maggioni M. 2019 Nonparametric inference of interaction laws in systems of agents from trajectory data. *Proceedings of the National Academy of Sciences* **116**, 14424–14433.
- Krizhevsky A, Sutskever I, Hinton GE. 2012 Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems* pp. 1097–1105.
- Bahdanau D, Cho K, Bengio Y. 2014 Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.
- Yang Y, Perdikaris P. 2019 Conditional deep surrogate models for stochastic, high-dimensional, and multi-fidelity systems. *Computational Mechanics* **64**, 417–434.

13. Han J, Jentzen A, Weinan E. 2018 Solving high-dimensional partial differential equations using deep learning. *Proceedings of the National Academy of Sciences* **115**, 8505–8510.
14. Rackauckas C, Ma Y, Martensen J, Warner C, Zubov K, Supekar R, Skinner D, Ramadhan A. 2020 Universal Differential Equations for Scientific Machine Learning. *arXiv preprint arXiv:2001.04385*.
15. Gholami A, Keutzer K, Biros G. 2019 Anode: Unconditionally accurate memory-efficient gradients for neural odes. *arXiv preprint arXiv:1902.10298*.
16. Chen TQ, Rubanova Y, Bettencourt J, Duvenaud DK. 2018 Neural ordinary differential equations. In *Advances in neural information processing systems* pp. 6571–6583.
17. Brunton SL, Proctor JL, Kutz JN. 2016 Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the National Academy of Sciences* **113**, 3932–3937.
18. Rudy SH, Brunton SL, Proctor JL, Kutz JN. 2017 Data-driven discovery of partial differential equations. *Science Advances* **3**, e1602614.
19. Brennan C, Venturi D. 2018 Data-driven closures for stochastic dynamical systems. *Journal of Computational Physics* **372**, 281–298.
20. Raissi M, Perdikaris P, Karniadakis GE. 2018 Multistep neural networks for data-driven discovery of nonlinear dynamical systems. *arXiv preprint arXiv:1801.01236*.
21. Qin T, Wu K, Xiu D. 2019 Data driven governing equations approximation using deep neural networks. *Journal of Computational Physics* **395**, 620–635.
22. Raissi M, Perdikaris P, Karniadakis GE. 2017 Inferring solutions of differential equations using noisy multi-fidelity data. *Journal of Computational Physics* **335**, 736–746.
23. Raissi M, Perdikaris P, Karniadakis GE. 2019 Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics* **378**, 686–707.
24. Zhu Y, Zabaras N, Koutsourelakis PS, Perdikaris P. 2019 Physics-constrained deep learning for high-dimensional surrogate modeling and uncertainty quantification without labeled data. *Journal of Computational Physics* **394**, 56–81.
25. Yang Y, Perdikaris P. 2019 Adversarial uncertainty quantification in physics-informed neural networks. *Journal of Computational Physics* **394**, 136–152.
26. Yang L, Zhang D, Karniadakis GE. 2020 Physics-Informed Generative Adversarial Networks for Stochastic Differential Equations. *SIAM Journal on Scientific Computing* **42**, A292–A317.
27. Wang S, Teng Y, Perdikaris P. 2020 Understanding and mitigating gradient pathologies in physics-informed neural networks. *arXiv preprint arXiv:2001.04536*.
28. Wang Q, Hesthaven JS, Ray D. 2019 Non-intrusive reduced order modeling of unsteady flows using artificial neural networks with application to a combustion problem. *Journal of computational physics* **384**, 289–307.
29. Deveney T, Mueller E, Shardlow T. 2019 A deep surrogate approach to efficient Bayesian inversion in PDE and integral equation models. *arXiv preprint arXiv:1910.01547*.
30. Raissi M, Karniadakis GE. 2018 Hidden physics models: Machine learning of nonlinear partial differential equations. *Journal of Computational Physics* **357**, 125–141.
31. Tartakovsky AM, Marrero CO, Perdikaris P, Tartakovsky GD, Barajas-Solano D. 2018 Learning parameters and constitutive relationships with physics informed deep neural networks. *arXiv preprint arXiv:1808.03398*.
32. Kissas G, Yang Y, Hwang E, Witschey WR, Detre JA, Perdikaris P. 2020 Machine learning in cardiovascular flows modeling: Predicting arterial blood pressure from non-invasive 4D flow MRI data using physics-informed neural networks. *Computer Methods in Applied Mechanics and Engineering* **358**, 112623.
33. Chen Y, Lu L, Karniadakis GE, Negro LD. 2019 Physics-informed neural networks for inverse problems in nano-optics and metamaterials. *arXiv preprint arXiv:1912.01085*.
34. Sahli Costabal F, Yang Y, Perdikaris P, Hurtado DE, Kuhl E. 2020 Physics-informed neural networks for cardiac activation mapping. *Frontiers in Physics* **8**, 42.
35. Neal RM et al.. 2011 MCMC using Hamiltonian dynamics. *Handbook of markov chain monte carlo* **2**, 2.
36. Pontryagin LS. 2018 *Mathematical theory of optimal processes*. Routledge.
37. van Merriënboer B, Breuleux O, Bergeron A, Lamblin P. 2018 Automatic differentiation in ML: Where we are and where we should be going. In *Advances in neural information processing systems* pp. 8757–8767.

38. Hastie T, Tibshirani R, Friedman J. 2009 *The elements of statistical learning: data mining, inference, and prediction*. Springer Science & Business Media.
39. Ghahramani Z. 2015 Probabilistic machine learning and artificial intelligence. *Nature* **521**, 452–459.
40. Gulrajani I, Ahmed F, Arjovsky M, Dumoulin V, Courville AC. 2017 Improved training of wasserstein gans. In *Advances in neural information processing systems* pp. 5767–5777.
41. Su W, Boyd S, Candes E. 2014 A differential equation for modeling NesterovâŽs accelerated gradient method: Theory and insights. In *Advances in Neural Information Processing Systems* pp. 2510–2518.
42. Bottou L. 2010 Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010* pp. 177–186. Springer.
43. Iserles A. 2008 *A First Course in the Numerical Analysis of Differential Equations*. Cambridge Texts in Applied Mathematics. Cambridge University Press 2 edition.
44. Williams PM. 1995 Bayesian regularization and pruning using a Laplace prior. *Neural computation* **7**, 117–143.
45. Geweke J. 1993 Bayesian treatment of the independent Student-t linear model. *Journal of applied econometrics* **8**, S19–S40.
46. Gelman A, Carlin JB, Stern HS, Dunson DB, Vehtari A, Rubin DB. 2013 *Bayesian data analysis*. CRC press.
47. Winkler RL. 1967 The assessment of prior distributions in Bayesian analysis. *Journal of the American Statistical association* **62**, 776–800.
48. Bernardo JM. 1979 Reference posterior distributions for Bayesian inference. *Journal of the Royal Statistical Society: Series B (Methodological)* **41**, 113–128.
49. Berger JO. 1990 Robust Bayesian analysis: sensitivity to the prior. *Journal of statistical planning and inference* **25**, 303–328.
50. Gilks WR, Best NG, Tan K. 1995 Adaptive rejection Metropolis sampling within Gibbs sampling. *Journal of the Royal Statistical Society: Series C (Applied Statistics)* **44**, 455–472.
51. Millar RB, Meyer R. 2000 Non-linear state space modelling of fisheries biomass dynamics by using Metropolis-Hastings within-Gibbs sampling. *Journal of the Royal Statistical Society: Series C (Applied Statistics)* **49**, 327–342.
52. Yazdani A, Raissi M, Karniadakis GE. 2019 Systems biology informed deep learning for inferring parameters and hidden dynamics. *bioRxiv* p. 865063.
53. Champion K, Lusch B, Kutz JN, Brunton SL. 2019 Data-driven discovery of coordinates and governing equations. *Proceedings of the National Academy of Sciences* **116**, 22445–22451.
54. Schnakenberg J. 1979 Simple chemical reaction systems with limit cycle behaviour. *Journal of theoretical biology* **81**, 389–400.
55. Glorot X, Bengio Y. 2010 Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics* pp. 249–256.
56. Kingma DP, Ba J. 2014 Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.
57. Hoffman MD, Gelman A. 2014 The No-U-Turn sampler: adaptively setting path lengths in Hamiltonian Monte Carlo.. *Journal of Machine Learning Research* **15**, 1593–1623.
58. Lusch B, Kutz JN, Brunton SL. 2018 Deep learning for universal linear embeddings of nonlinear dynamics. *Nature communications* **9**, 4950.
59. Li X, Wong TKL, Chen RT, Duvenaud D. 2020 Scalable Gradients for Stochastic Differential Equations. *arXiv preprint arXiv:2001.01328*.