Variational multiple shooting for Bayesian ODEs with Gaussian processes

Abstract

Recent machine learning advances have proposed black-box estimation of *unknown continuous-time system dynamics* directly from data. However, earlier works are based on approximative solutions or point estimates. We propose a novel Bayesian nonparametric model that uses Gaussian processes to infer posteriors of unknown ODE systems directly from data. We derive sparse variational inference with decoupled functional sampling to represent vector field posteriors. We also introduce a probabilistic shooting augmentation to enable efficient inference from arbitrarily long trajectories. The method demonstrates the benefit of computing vector field posteriors, with predictive uncertainty scores outperforming alternative methods on multiple ODE learning tasks.

1 INTRODUCTION

Ordinary differential equations (ODEs) are powerful models for continuous-time non-stochastic systems, which are ubiquitous from physical and life sciences to engineering (Hirsch et al., 2012). In this work, we consider non-linear ODE systems

$$\mathbf{x}(t) = \mathbf{x}_0 + \int_0^t \mathbf{f}(\mathbf{x}(\tau))d\tau \tag{1}$$

$$\dot{\mathbf{x}}(t) := \frac{d\mathbf{x}(t)}{dt} = \mathbf{f}(\mathbf{x}(t)), \tag{2}$$

where the state vector $\mathbf{x}(t) \in \mathbb{R}^D$ evolves over time $t \in \mathbb{R}_+$ from an initial state $\mathbf{x}_0$ following its time derivative $\dot{\mathbf{x}}(t)$, and τ is an auxiliary time variable. Our goal is to learn the differential function $\mathbf{f} : \mathbb{R}^D \mapsto \mathbb{R}^D$ from state observations, when the functional form of $\mathbf{f}$ is unknown.

The conventional mechanistic approach involves manually defining the equations of dynamics and optimizing their parameters (Butcher and Goodwin, 2008), or inferring their posteriors (Girolami, 2008) from data. However, the equations are unknown or ambiguous for many systems, such as human motion (Wang et al., 2008). Some early works explored fitting unknown ODEs with splines (Henderson and Michailidis, 2014), Gaussian processes (Äijö and Lähdesmäki, 2009) or kernel methods (Heinonen and d'Alché-Buc, 2014) by resorting to less accurate gradient matching approximations (Varah, 1982). Recently, Heinonen et al. (2018) proposed estimation of free-form non-linear dynamics using Gaussian processes without gradient matching. However, the approach is restricted to learning point estimates of the dynamics, limiting the uncertainty characterization and generalization. Chen et al. (2018) proposed modeling ODEs with neural networks and adjoints, which was later extended to the Bayesian setting by Dandekar et al. (2020). However, the gradient descent training in such approaches can be ill-suited for complex or long-horizon ODEs with typically highly non-linear integration maps (Diehl and Gros, 2017).

In this work, we introduce efficient Bayesian learning of unknown, non-linear ODEs. Our contributions are:

- We introduce a way of learning posteriors of vectorfields using Gaussian processes as flexible priors over differentials $\mathbf{f}$, and thereby build on the work by Heinonen et al. (2018). We adapt decoupled functional sampling to simulate ODEs from vector field posteriors.
- For the difficult problem of gradient optimizations of ODEs, we introduce a novel probabilistic shooting method. It is motivated by the canonical shooting methods from optimal control and makes inference stable and efficient on long trajectories.
- We empirically show the effectiveness of the proposed method even while learning from a limited number of observations. We demonstrate the ability to infer arbitrarily long trajectories efficiently with the shooting extension.

Submitted to the 38th Conference on Uncertainty in Artificial Intelligence (UAI 2022). **To be used for reviewing only**.

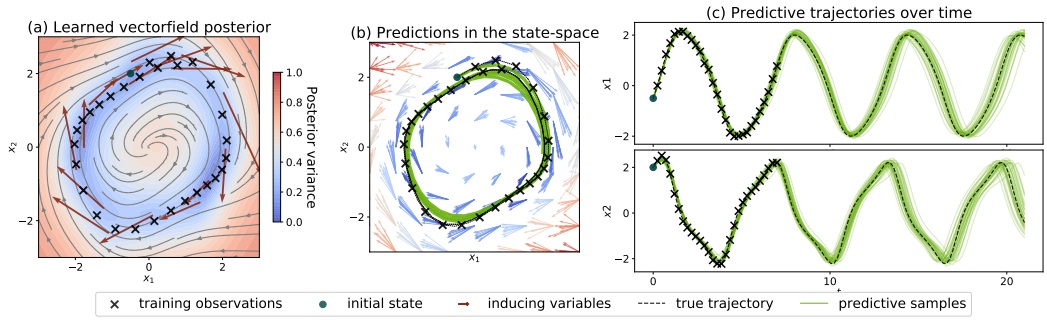


Figure 1: Illustration of GPODE: The model learns a GP posterior (a) of a vector field. Valid ODE trajectories are sampled from the posterior process as shown in (b) and (c).

2 RELATED WORKS

Mechanistic ODE models. In mechanistic modelling the equation $\mathbf{f}_{\boldsymbol{\theta}}$ is predefined with a set of coefficients $\boldsymbol{\theta}$ to be fitted (Butcher and Goodwin, 2008). Several works have proposed embedding mechanistic models within Bayesian or Gaussian process models (Calderhead et al., 2008; Dondelinger et al., 2013; Wenk et al., 2020). Recently both Julia and Stan have introduced support for Bayesian analysis of parametric ODEs (Rackauckas and Nie, 2017; Stan, 2021). Since this line of work assumes a known dynamics model, we do not consider these methods in the experiments.

Free-form ODE models. Multiple works have proposed fitting unknown, non-linear and free-form ODE differentials with gradient matching using splines (Ramsay et al., 2007), Gaussian processes (Äijö and Lähdesmäki, 2009) or kernel methods (Heinonen and d'Alché-Buc, 2014). Recently, Heinonen et al. (2018) proposed accurate *maximum a posteriori*(MAP) optimisation of vector fields with sensitivity equation gradients (Kokotovic and Heller, 1967). Neural ODEs (Chen et al., 2018) introduced adjoint gradients (Pontryagin et al., 1962) along with flexible black-box neural network vector fields. Several extensions to learning latent ODEs have been proposed (Yildiz et al., 2019; Rubanova et al., 2019).

Discrete-time state-space models. There is a large literature on Markovian state-space models that operate over discrete time increments (Wang et al., 2005; Turner et al., 2010; Frigola et al., 2014). Typically nonlinear state transition functions are modeled with Gaussian processes and applied to latent state estimation or system identification problems with dynamical systems (Eleftheriadis et al., 2017; Doerr et al., 2018; Ialongo et al., 2019). In this paper, we focus strictly on continuous-time models and leave the study of discrete vs. continuous formulations for future work.

Stochastic differential equations. As an alternative formulation of inferring unknown dynamics from observational data, one can assume stochastic transitions and learn models of stochastic differential equations (SDEs). Existing works have utilized Gaussian processes (Archambeau et al., 2007; Duncker et al., 2019; Jørgensen et al., 2020) and neural networks (Tzen and Raginsky, 2019; Li et al., 2020) to model non-linear SDEs. However, since they assume a different model (i.e. deterministic transitions vs stochastic transitions), we will restrict the experimental comparisons to other ODE-based approaches.

3 METHODS

We consider the problem of learning ODEs (2) with GPs and propose a Bayesian model to infer posteriors over the differential $\mathbf{f}(\cdot)$.

3.1 BAYESIAN MODELING OF ODES USING GPS

We assume a sequence of N observations $\mathbf{Y} = (\mathbf{y}_1, \mathbf{y}_2, \dots \mathbf{y}_N)^T \in \mathbb{R}^{N\times D}$ along a trajectory, with $\mathbf{y}_i \in \mathbb{R}^D$ representing the noisy observation of the unknown state $\mathbf{x}(t_i) \in \mathbb{R}^D$ at time t_i. Similar to Heinonen et al. (2018), we assume a zero mean vector-valued Gaussian process prior over $\mathbf{f}$,

$$\mathbf{f}(\mathbf{x}) \sim \mathcal{GP}(\mathbf{0}, K(\mathbf{x}, \mathbf{x}')), \tag{3}$$

which defines a distribution of differentials $\mathbf{f}(\mathbf{x})$ with covariance $\operatorname{cov}[\mathbf{f}(\mathbf{x}), \mathbf{f}(\mathbf{x}')] = K(\mathbf{x}, \mathbf{x}')$, where $K(\mathbf{x}, \mathbf{x}') \in \mathbb{R}^{D\times D}$ is a stationary matrix-valued kernel. We follow the commonly used sparse inference framework for GPs using inducing variables (Titsias, 2009), and augment the full model with inducing values $\mathbf{U} = (\mathbf{u}_1, \dots, \mathbf{u}_M)^T \in \mathbb{R}^{M\times D}$ and inducing locations $\mathbf{Z} = (\mathbf{z}_1, \dots, \mathbf{z}_M)^T \in \mathbb{R}^{M\times D}$ such that $\mathbf{u}_m = \mathbf{f}(\mathbf{z}_m)$. The inducing variables are trainable 'landmark' state-differential pairs, from which the rest of the differential field is interpolated (See Figure 1, where arrow locations are the $\mathbf{z}_m$ and arrow end-points are the $\mathbf{u}_m$). The inducing augmentation leads to the following prior and

conditionals (Hensman et al., 2013):

$$p(\mathbf{U}) = \mathcal{N}(\mathbf{U}|\mathbf{0}, \mathbf{K}_{\mathbf{ZZ}}), \tag{4}$$

$$p(\mathbf{f}|\mathbf{U};\mathbf{Z}) = \mathcal{N}(\mathbf{f}|\mathbf{A}\mathrm{vec}(\mathbf{U}), \mathbf{K}_{\mathbf{XX}} - \mathbf{A}\mathbf{K}_{\mathbf{ZZ}}\mathbf{A}^T), \tag{5}$$

where $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \dots \mathbf{x}_{N'})^T \in \mathbb{R}^{N' \times D}$ collects all the intermediate state evaluations $\mathbf{x}(t_i)$ encountered along a numerical approximation of the true continuous ODE integral (1), $\mathbf{f} = (\mathbf{f}(\mathbf{x}_1)^T, \dots, \mathbf{f}(\mathbf{x}'_N)^T)^T \in \mathbb{R}^{N'D \times 1}$, $\mathbf{K}_{\mathbf{XX}}$ is a block-partitioned matrix of size $N'D \times N'D$ with $D \times D$ blocks, so that block $(\mathbf{K}_{\mathbf{XX}})_{i,j} = K(\mathbf{x}_i, \mathbf{x}_j)$, and $\mathbf{A} = \mathbf{K}_{\mathbf{XZ}}\mathbf{K}_{\mathbf{ZZ}}^{-1}$. For notational simplicity, we assume that the measurement time points are among the time points of the intermediate state evaluations of a numerical ODE solver.

The joint probability distribution follows

$$p(\mathbf{Y}, \mathbf{f}, \mathbf{U}, \mathbf{x}_0) = p(\mathbf{Y}|\mathbf{f}, \mathbf{x}_0)p(\mathbf{f}, \mathbf{U})p(\mathbf{x}_0) \tag{6}$$

$$= \prod_{i=1}^{N} p(\mathbf{y}_i|\mathbf{f}, \mathbf{x}_0)p(\mathbf{f}|\mathbf{U})p(\mathbf{U})p(\mathbf{x}_0), \tag{7}$$

where the conditional distribution $p(\mathbf{y}_i|\mathbf{f}, \mathbf{x}_0) = p(\mathbf{y}_i|\mathbf{x}_i)$ computes the likelihood over ODE state solutions $\mathbf{x}_i = \mathbf{x}_0 + \int_0^{t_i} \mathbf{f}(\mathbf{x}(\tau))d\tau$.

3.2 VARIATIONAL INFERENCE FOR GP-ODES

In contrast to earlier approach that estimates MAP solutions (Heinonen et al., 2018), our goal is to infer the posterior distribution $p(\mathbf{f}, \mathbf{x}_0|\mathbf{Y})$ of the vector field $\mathbf{f}$ and initial value $\mathbf{x}_0$ from observations $\mathbf{Y}$. The posterior is intractable due to the non-linear integration map $\mathbf{x}_0 \overset{\mathbf{f}}{\mapsto} \mathbf{x}(t)$.

We use the stochastic variational inference (SVI) formulation for sparse GPs (Hensman et al., 2013) in this work. We introduce a factorized Gaussian posterior approximation for the inducing variables across state dimensions $q(\mathbf{U}) = \prod_{d=1}^{D} \mathcal{N}(\mathbf{u}_d|\mathbf{m}_d, \mathbf{Q}_d), \mathbf{u}_d \in \mathbb{R}^M$ where $\mathbf{m}_d \in \mathbb{R}^M, \mathbf{Q}_d \in \mathbb{R}^{M \times M}$ are the mean and covariance parameters of the variational Gaussian posterior approximation for the inducing variables. We treat the inducing locations $\mathbf{Z}$ as optimized hyperparameters. The posterior distribution for the variational approximation can be written as

$$q(\mathbf{f}) = \int p(\mathbf{f}|\mathbf{U})q(\mathbf{U})d\mathbf{U} \tag{8}$$

$$= \int \mathcal{N}\left(\mathbf{f}|\mathbf{A}\mathrm{vec}(\mathbf{U}), \mathbf{K}_{\mathbf{XX}} - \mathbf{A}\mathbf{K}_{\mathbf{ZZ}}\mathbf{A}^T\right) q(\mathbf{U})d\mathbf{U}. \tag{9}$$

The posterior inference goal then translates to estimating the posterior $p(\mathbf{f}, \mathbf{U}, \mathbf{x}_0|\mathbf{Y})$ of the inducing points $\mathbf{U}$ and initial state $\mathbf{x}_0$. Under variational inference this learning objective

$$\arg\min_q \ \mathrm{KL}\left[\, q(\mathbf{f}, \mathbf{U}, \mathbf{x}_0) \,||\, p(\mathbf{f}, \mathbf{U}, \mathbf{x}_0|\mathbf{Y}) \,\right] \tag{10}$$

translates into maximizing the evidence lowerbound (ELBO),

$$\log p(\mathbf{Y}) \geq \sum_{i=1}^{N} \overbrace{\mathbb{E}_{q(\mathbf{f},\mathbf{x}_0)} \log p(\mathbf{y}_i|\mathbf{f}, \mathbf{x}_0)}^{\text{variational likelihood}} - \overbrace{\mathrm{KL}[q(\mathbf{U})||p(\mathbf{U})]}^{\text{inducing KL}}$$
$$- \underbrace{\mathrm{KL}[q(\mathbf{x}_0)||p(\mathbf{x}_0)]}_{\text{initial state KL}}, \tag{11}$$

where we also assume variational approximation $q(\mathbf{x}_0) = \mathcal{N}(\mathbf{a}_0, \Sigma_0)$ for the initial state $\mathbf{x}_0$. See supplementary section 1.1 for detailed derivations of the above equations.

3.3 SAMPLING ODES FROM GAUSSIAN PROCESSES

The Picard-Lindelöf theorem (Lindelöf, 1894) ensures valid ODE systems define unique solutions to the initial value problem (IVP) (1). In order to sample valid state trajectories for the IVP, we need to efficiently sample GP functions $\mathbf{f}(\cdot) \sim q(\mathbf{f})$ (9). This way, we can evaluate the sample function $\mathbf{f}(\mathbf{x}(t))$ at arbitrary states $\mathbf{x}(t)$ encountered during ODE forward integration, while accounting for both the inducing and interpolation distributions of Equation (9). Unfortunately, function-space sampling of such GPs has prohibitive cubic complexity (Rasmussen and Williams, 2006; Ustyuzhaninov et al., 2020), while the more efficient weight-space sampling with Fouriers cannot accurately express the posterior (9) (Wilson et al., 2020).

We use the decoupled sampling that decomposes the posterior into two parts (Wilson et al., 2020),

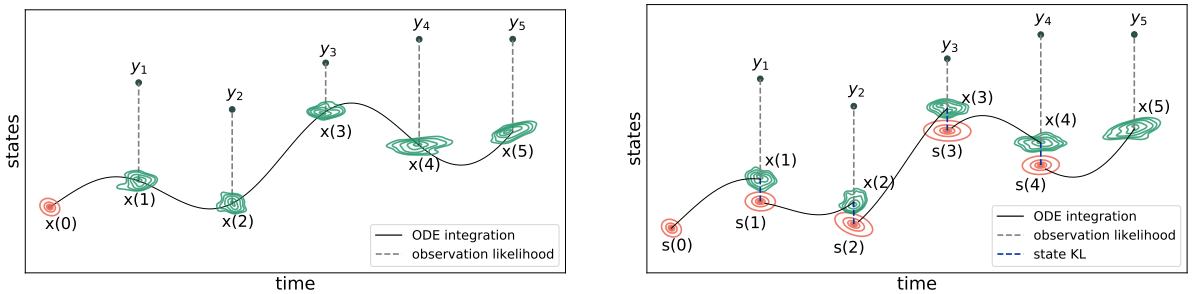
$$\overbrace{\mathbf{f}(\mathbf{x})|\mathbf{U}}^{\text{posterior}} = \overbrace{\mathbf{f}(\mathbf{x})}^{\text{prior}} + \overbrace{K(\mathbf{x}, \mathbf{Z})K(\mathbf{Z}, \mathbf{Z})^{-1}(\mathbf{U} - \mathbf{f}_{\mathbf{Z}})}^{\text{update}}. \tag{12}$$

$$\approx \sum_{i=1}^{S} \mathbf{w}_i \boldsymbol{\phi}_i(\mathbf{x}) + \sum_{j=1}^{M} \boldsymbol{\nu}_j K(\mathbf{x}, \mathbf{z}_j), \tag{13}$$

where we use S Fourier bases $\boldsymbol{\phi}_i(\cdot)$ with $\mathbf{w}_i \sim \mathcal{N}(\mathbf{0}, I)$ (Rahimi and Recht, 2007) to represent the stationary prior, and function basis $K(\cdot, \mathbf{z}_j)$ for the posterior update with $\boldsymbol{\nu} = K(\mathbf{Z}, \mathbf{Z})^{-1}(\mathbf{U} - \boldsymbol{\Phi}\mathbf{W})$, $\boldsymbol{\Phi} = \boldsymbol{\phi}(\mathbf{Z}) \in \mathbb{R}^{M \times S}, \mathbf{W} \in \mathbb{R}^{S \times D}$. By combining these two steps, we can accurately evaluate functions from the posterior (9) in linear time at arbitrary locations. We refer the reader to the supplementary section 1.2 for more details. We note that concurrent works by Mikheeva et al. (2021) and Ensinger et al. (2021) also utilize the decoupled-sampling to infer ODE posteriors with GPs.

3.4 AUGMENTING THE ODE MODEL WITH SHOOTING SYSTEM

A key bottleneck in ODE modeling is the poor gradient descent performance over long integration times $\mathbf{x}_{0:T}$, which



(a) The full model formulation

(b) Shooting augmentation

Figure 2: Illustrations of GPODE formulations: the full model formulation (a) follows the long trajectory integration, whereas the shooting version (b) splits the long trajectory into multiple short subintervals.

can exhibit vanishing or exploding gradients (Haber and Ruthotto, 2017; Choromanski et al., 2020). Earlier approaches tackled this issue mainly with more accurate numerical solvers (Zhuang et al., 2020, 2021). The nonlinearity of the integration map $\mathbf{x}_0 \overset{\mathbf{f}}{\mapsto} \mathbf{x}_t$ motivates us to instead segment the full integration $\mathbf{x}_{0:T}$ into short segments, which are easier to optimize and can be trivially parallelized. This is called the *multiple shooting* method in optimal control literature (Osborne, 1969; Bock and Plitt, 1984), in the context of parameter estimation of ODEs (vanDomselaar and Hemker, 1975; Bock, 1983). Recently, Massaroli et al. (2021); Turan and Jäschke (2021) also introduced a multiple-shooting framework within the context of deterministic neural ODEs. We introduce probabilistic shooting for the Gaussian process posterior inference of ODEs.

We begin by introducing shooting state variables $\mathbf{S} = (\mathbf{s}_0, \mathbf{s}_1, \ldots, \mathbf{s}_{N-1})$, $\mathbf{s}_i \in \mathbb{R}^D$, and segment the continuous state function $\mathbf{x}(t; \mathbf{x}_0)$ (1) into N segments $\{(\mathbf{s}_{i-1}, \mathbf{x}(t_i; \mathbf{s}_{i-1}))\}_{i=1}^N$ that branch from the shooting variables $\mathbf{s}_{i-1}$ (See Figure 2);

$$\mathbf{x}(t_i; \mathbf{s}_{i-1}) = \mathbf{s}_{i-1} + \int_{t_{i-1}}^{t_i} \mathbf{f}(\mathbf{x}(\tau))d\tau. \tag{14}$$

In addition, every shooting variable is approximately matched with the ODE state evolution from the previous shooting state,

$$\mathbf{s}_i = \mathbf{x}(t_i; \mathbf{s}_{i-1}) + \boldsymbol{\xi}, \tag{15}$$

where $\boldsymbol{\xi} \in \mathbb{R}^D$ represents the tolerance parameter controlling the shooting approximation. The augmented system is equivalent to the original ODE system in case the constraints $\mathbf{s}_i = \mathbf{x}(t_i; \mathbf{s}_{i-1})$ are satisfied exactly at the limit $\boldsymbol{\xi} \to \mathbf{0}$. We place a Gaussian prior over the tolerance parameter $\boldsymbol{\xi} \sim \mathcal{N}(\mathbf{0}, \sigma_\xi^2 \mathbf{I})$, which translates into the following prior over shooting variables

$$p(\mathbf{s}_i | \mathbf{s}_{i-1}) = \mathcal{N}(\mathbf{s}_i | \mathbf{x}(t_i; \mathbf{s}_{i-1}), \sigma_\xi^2 \mathbf{I}). \tag{16}$$

Further, the joint probability of the augmented model after placing a GP prior over the vectorfield $\mathbf{f}$ can be written as

$$p(\mathbf{Y}, \mathbf{S}, \mathbf{f}) = \prod_{i=1}^{N} p(\mathbf{y}_i | \mathbf{s}_{i-1}, \mathbf{f}) \prod_{i=1}^{N-1} p(\mathbf{s}_i | \mathbf{s}_{i-1}, \mathbf{f}) p(\mathbf{s}_0) p(\mathbf{f}). \tag{17}$$

3.5 VARIATIONAL INFERENCE FOR THE AUGMENTED MODEL

To infer the augmented posterior $p(\mathbf{f}, \mathbf{U}, \mathbf{S} | \mathbf{Y})$ we introduce variational approximation for the shooting variables $q(\mathbf{S}) = q(\mathbf{s}_0) \cdots q(\mathbf{s}_{N-1})$, where each distribution $q(\mathbf{s}_i) = \mathcal{N}(\mathbf{s}_i | \mathbf{a}_i, \Sigma_i)$ is a Gaussian. This results in the joint variational approximation

$$q(\mathbf{S}, \mathbf{f}, \mathbf{U}) = \prod_{i=0}^{N-1} q(\mathbf{s}_i) p(\mathbf{f} | \mathbf{U}) q(\mathbf{U}), \tag{18}$$

and the following evidence lower bound for the shooting model,

$$\begin{aligned} \mathcal{L}_{\text{shooting}} &= \sum_{i=1}^{N} \mathbb{E}_{q(\mathbf{s}_{i-1}, \mathbf{f})} \Big[\log p(\mathbf{y}_i | \mathbf{s}_{i-1}, \mathbf{f}) \Big] \\ &+ \sum_{i=1}^{N-1} \mathbb{E}_{q(\mathbf{s}_i, \mathbf{s}_{i-1}, \mathbf{f})} \Big[\log p\left(\mathbf{s}_i | \mathbf{s}_{i-1}, \mathbf{f}\right) \Big] - \mathbb{E}_{q(\mathbf{s}_i)} \Big[\log q(\mathbf{s}_i) \Big] \\ &- \mathrm{KL}[q(\mathbf{s}_0) \,||\, p(\mathbf{s}_0)] - \mathrm{KL}[q(\mathbf{U}) \,||\, p(\mathbf{U})]. \end{aligned} \tag{19}$$

The ELBO consists of an expected log-likelihood term, which matches the state evolution (14) from every shooting variable to the corresponding observation. In addition, the posterior approximation for every shooting variable is also matched with the ODE evolution of the approximated posterior of the previous shooting state, leading to corresponding cross-entropy and entropy terms.

The ELBO for the augmented shooting model requires solving only the short segments (14) with simpler integration maps, thus great at mitigating problems with vanishing/exploring gradients. Since the involved numerical ODE

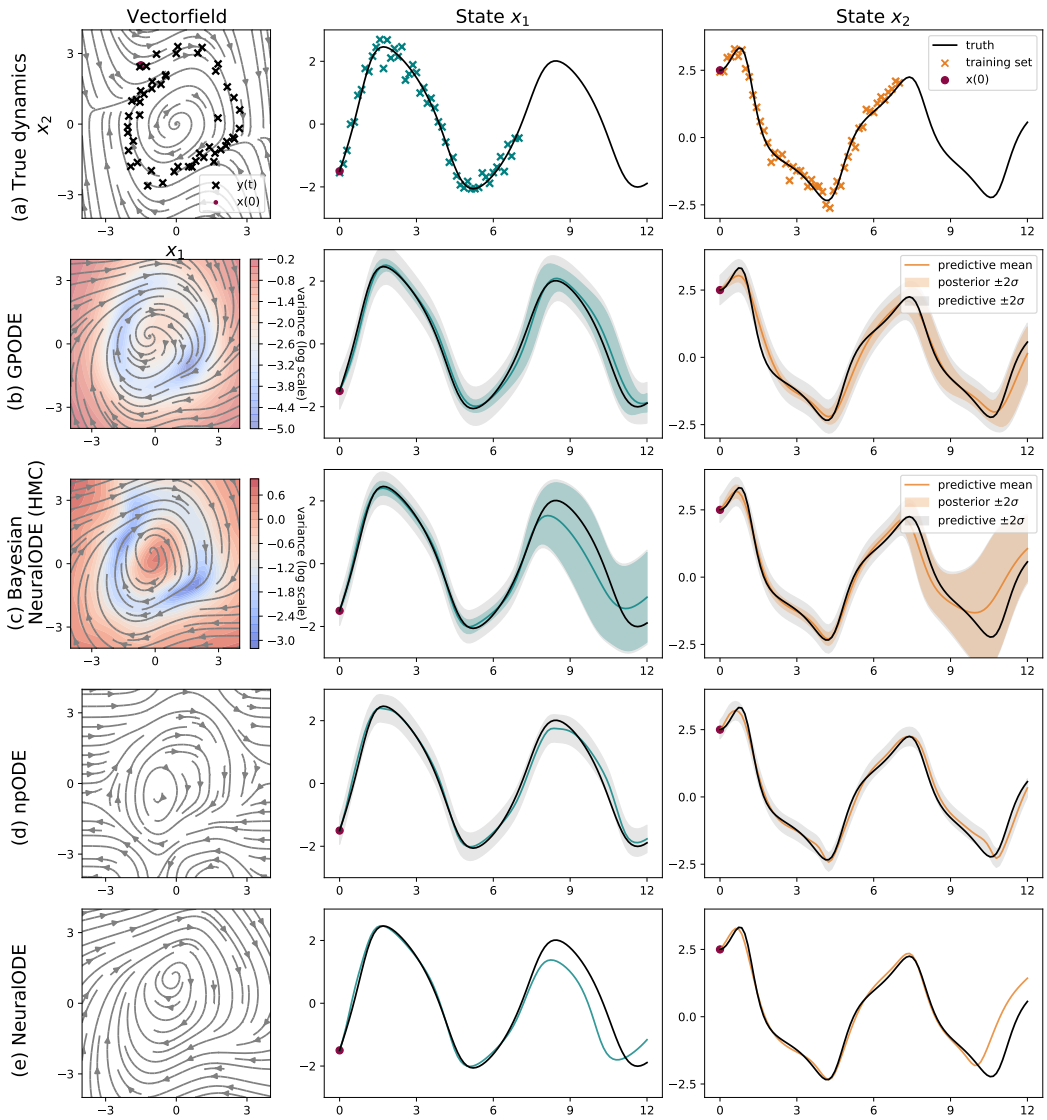


Figure 3: Learning the 2D Van der Pol dynamics **(a)** with alternative methods **(b-d)**. Column 1 shows the vector fields while columns 2 and 3 show the state trajectories $x_1(t)$ and $x_2(t)$. GPODE learns the posterior accurately.

integration can be done in parallel, the shooting model is also computationally faster than the full model in practice. See supplementary section 1.3 for a plate diagram and detailed derivation of the approach.

4 EXPERIMENTS

We validate the proposed method on Van der Pol (VDP) and FitzHugh–Nagumo (FHN) systems and on the task of learning human motion dynamics (MoCap). The predictive performance of the proposed GPODE is compared against npODE (Heinonen et al., 2018), NeuralODE (Chen et al., 2018) and Bayesian version of NeuralODE (Dandekar et al., 2020). We use 16 inducing points in VDP and FHN experiments and 100 inducing points for the MoCap experiments. Except for the NeuralODE model, we assume Gaussian observation likelihood, and infer the unknown noise scale parameter from the training data. All the experiments use squared exponential kernel with ARD. Along with the variational parameters, kernel lengthscales, signal variance, noise scale, and inducing locations are jointly optimized against the model ELBO while training. In addition, for the shooting model, we fix the constraint tolerance parameter to a small value $\sigma_\xi^2 = 1e^{-6}$ consistently across all the experiments. An open-source codebase for reproducing all the experiments will be provided upon acceptance.

We use the implicit `dopri5` solver with tolerance parameters `rtol`$= 1e^{-5}$ and `atol`$= 1e^{-5}$, and use the adjoint method for computing loss gradients with `torchdiffeq`[1]

[1] `https://github.com/rtqichen/torchdiffeq`

Table 1: VDP system learning performance on extrapolation task with observations on regular (task 1) and irregular time intervals (task 2). We report mean $\pm$ standard error over 5 runs from different random initialization, the best values bolded. (↑): higher is better, (↓) lower is better

	Task 1: Regular time-grid		Task 2: Irregular time-grid	
	MNLL (↓)	MSE (↓)	MNLL (↓)	MSE (↓)
Bayesian NeuralODE (HMC)	0.82 ± 0.01	1.45 ± 0.04	0.88 ± 0.01	1.68 ± 0.04
NeuralODE	-	0.29 ± 0.11	-	0.55 ± 0.07
npODE	1.47 ± 0.59	0.16 ± 0.05	8.89 ± 3.06	2.08 ± 0.78
GPODE	$\mathbf{0.60 \pm 0.03}$	$\mathbf{0.13 \pm 0.01}$	$\mathbf{0.41 \pm 0.18}$	$\mathbf{0.21 \pm 0.07}$

package (Chen et al., 2018). All the experiments are repeated 5 times with random initialization, and means and standard errors are reported over multiple runs. The predictive performance of different models are measured with mean squared error (MSE) and mean negative log likelihood (MNLL) metrics.

4.1 LEARNING VAN DER POL DYNAMICS

We first illustrate the effectiveness of the proposed method by inferring the vector field posterior on a two-dimensional VDP (see Figure 3),

$$\begin{aligned} \dot{x}_1 &= x_2, \\ \dot{x}_2 &= -x_1 + 0.5x_2(1 - x_1^2). \end{aligned} \tag{20}$$

We simulate a trajectory of 50 states following the true system dynamics from the initial state $(x_1(0), x_2(0)) = (-1.5, 2.5)$, and add Gaussian noise with $\sigma^2 = 0.05$ to generate the training data. We explore two scenarios with training time interval $t \in [0, 7]$ and forecasting interval $t \in [7, 14]$: (1) over a regularly sampled time grid, (2) over an irregular grid using uniform random sampling of time points. Task (2) demonstrates one of the key advantages of continuous-time models with the ability to handle irregular data.

Figure 3(b) shows that both GPODE and Bayesian NeuralODE learn a vector field posterior whose posterior mean closely matches the ground truth, with low variance (blue regions) near the observed data. The posterior variance increases away from the observed data (orange regions), indicating a good uncertainty characterization, while the npODE with MAP estimation seems to overfit. NeuralODE learns an appropriate vector field, but requires careful tuning of regularization and hyperparameters for a good fit with a limited number of observations. A quantitative evaluation of the model fits in Table 1 indicates the better performance of GPODE as compared to the other methods under comparison.

Table 2: Imputation results on the FHN system.

	MNLL (↓)	MSE (↓)
Bayesian NeuralODE (HMC)	0.77 ± 0.12	0.24 ± 0.03
NeuralODE	-	0.18 ± 0.00
npODE	6.49 ± 1.49	$\mathbf{0.08 \pm 0.01}$
GPODE	$\mathbf{0.09 \pm 0.05}$	$\mathbf{0.07 \pm 0.02}$

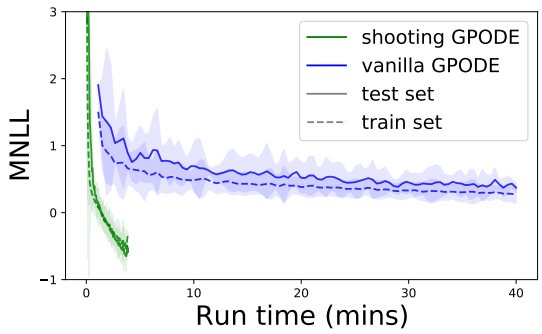


Figure 4: Optimization efficiency with GPODE models.

4.2 LEARNING WITH MISSING OBSERVATIONS

We illustrate the usefulness of learning Bayesian ODE posteriors under missing data with the FHN oscillator

$$\begin{aligned} \dot{x}_1 &= 3(x_1 - x_1^3/3 + x_2), \\ \dot{x}_2 &= (0.2 - 3x_1 - 0.2x_2)/3. \end{aligned} \tag{21}$$

We generate a training sequence by simulating 25 regularly-sampled time points from $t \in [0, 5.0]$ with added Gaussian noise with $\sigma^2 = 0.025$. We remove all observations at the quadrant $x_1 > 0, x_2 < 0$ and evaluate model accuracy in this region. The interpolation performance for different models is shown in Table 2. The point estimates of npODE and NeuralODE have biases, while the Bayesian variants of GPODE and NeuralODE provide good uncertainty estimates corresponding to their better predictive performance.

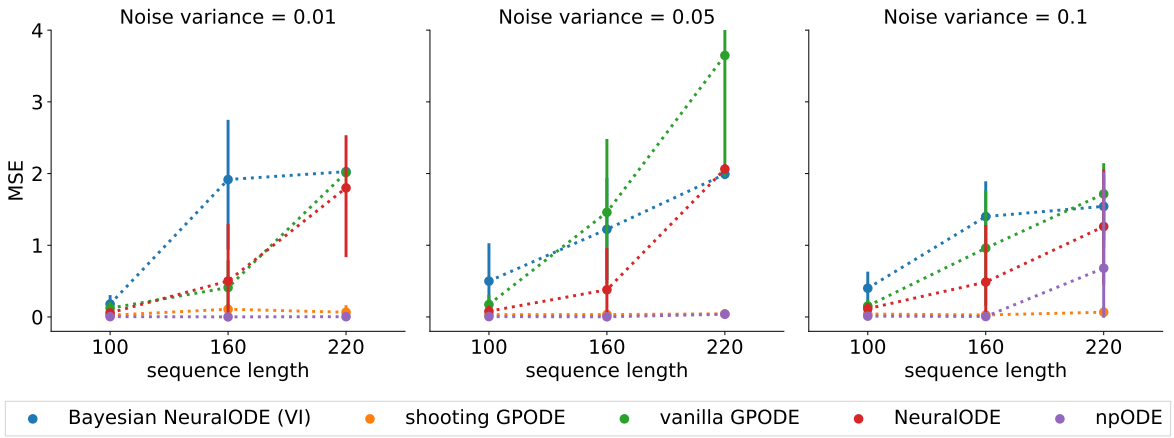


Figure 5: Varying sequence length and observation noise: shooting formulation makes GPODE feasible for long sequences, outperforming the non-shooting version and competing methods. We report the results for different levels of observation noise and training sequence length on the VDP system.

4.3 LEARNING LONG TRAJECTORIES WITH THE SHOOTING FORMULATION

We demonstrate the necessity of the shooting formulation for working with long training trajectories. We use the VDP system with four observations per unit of time for $T = (25, 40, 55)$ corresponding to $N = (100, 160, 220)$ observed states. We also vary the observation variance as $\sigma^2 = (0.01, 0.05, 0.1)$ and test the model for forecasting additional 50 time points.

Figure 5 demonstrates that vanilla-GPODE and NeuralODE, and Bayesian NeuralODE fail to fit the data with long sequences on all noise levels. In contrast, inference for the shooting model is successful in all settings. The npODE is remarkably robust to long trajectories.

Figure 4 shows a runtime trace comparison between vanilla GPODE and the shooting variant in wall-clock time for a fixed budget of 2000 optimization steps on the VDP system with $N = 100$, $T = 25$ and $\sigma^2 = 0.01$. The shooting model converges approximately 10 times faster. The speedup stems from the parallelization of the shooting ODE solver, since the shooting method splits the full IVP problem into numerous short and less non-linear IVPs. In addition, the shooting method relaxes the inference problem with its auxiliary augmentation. This experiment was conducted on a system with AMD Ryzen 5 3600 processor and Nvidia GeForce GTX 1660S GPUs.

4.4 LEARNING HUMAN MOTION DYNAMICS

We learn the dynamics of human motion from noisy experimental data from CMU MoCap database for three subjects, `09`, `35` and `39`. The dataset consists of 50 sensor readings from different parts of the body while walking or running. We follow the preprocessing of Wang et al. (2008) and center the data. The dataset was further split into train, test, and validation sequences. We observed that the NeuralODE, the Bayesian NeuralODE version with VI, and npODE models suffer from over-fitting, and we remedy this by applying early stopping by monitoring the validation loss during optimization.

We project the original 50-dimensional data into a 5-dimensional latent space using PCA and learn the dynamics in the latent space. To compute the data likelihood, we project the latent dynamics back to the original data space by inverting the PCA. We divide the experiment into sub-tasks MoCap-short and MoCap-long, based on the length of the sequence considered for model training (see the supplementary section for more details on the dataset and experimental setup). The model predictive performance is measured on unseen test sequences in both tasks.

Table 3 indicates that GPODE outperforms the competing npODE and NeuralODE model variants. Figure 6 visualizes the predicted dynamics for a test sequence. The GPODE variants have reasonable posterior uncertainties, while NeuralODE variants and npODE tend to be overconfident and make more mistakes (see Figure 6 (b), sensors `05`, `41` and `47`) . We note that some variations in the data space cannot be accurately estimated due to the low-dimensional PCA projection.

5 CONCLUSION AND DISCUSSION

We proposed a novel model for Bayesian inference of ODEs using Gaussian processes. With this approach, one can model unknown ODE systems directly from the observational data and learn posteriors of the continuous-time vector fields. In contrast, earlier works produce point estimate solutions. We believe this to be a significant addition to the data-descriptive ODE modeling methods, especially for ap-

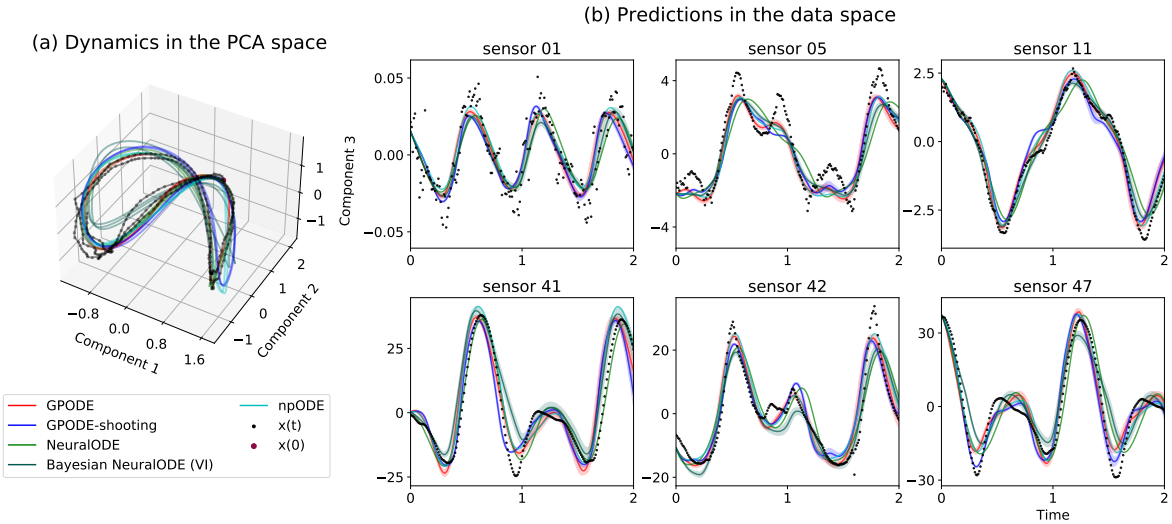


Figure 6: Learning the walking dynamics of subject 39: The true dynamics and predicted dynamics (mean) for the first three components in PCA space are shown in (a). Corresponding trajectories in the observation space for 6 different sensors are shown in (b) (We do not plot the observation noise variance)

Table 3: Test MNLL and MSE metrics for dynamics prediction task on CMU MoCap dataset.

Metric	Method	Subject 09		Subject 35		Subject 39	
		short	long	short	long	short	long
MNLL(↓)	Bayesian NeuralODE (VI)	2.03 ± 0.10	1.50 ± 0.05	1.42 ± 0.05	1.37 ± 0.06	1.61 ± 0.07	1.45 ± 0.03
	npODE	2.09 ± 0.01	1.78 ± 0.08	1.67 ± 0.02	1.66 ± 0.04	2.06 ± 0.05	1.78 ± 0.04
	GPODE-vanilla	1.30 ± 0.02	1.26 ± 0.02	1.27 ± 0.04	1.39 ± 0.04	1.29 ± 0.01	$\mathbf{1.13 \pm 0.01}$
	GPODE-shooting	$\mathbf{1.19 \pm 0.02}$	$\mathbf{1.14 \pm 0.02}$	$\mathbf{1.25 \pm 0.06}$	$\mathbf{1.08 \pm 0.02}$	$\mathbf{1.25 \pm 0.01}$	1.36 ± 0.02
MSE(↓)	Bayesian NeuralODE (VI)	25.50 ± 1.70	21.32 ± 2.58	23.09 ± 3.95	20.86 ± 2.95	53.34 ± 5.31	39.66 ± 6.82
	NeuralODE	27.53 ± 2.87	33.83 ± 2.46	36.50 ± 3.86	23.54 ± 0.56	115.38 ± 10.96	53.51 ± 2.98
	npODE	17.91 ± 1.62	19.76 ± 4.29	26.24 ± 2.88	22.83 ± 3.91	92.80 ± 15.74	55.94 ± 4.63
	GPODE-vanilla	15.78 ± 0.67	12.62 ± 1.14	16.14 ± 0.99	15.53 ± 0.76	$\mathbf{20.71 \pm 1.25}$	23.64 ± 1.94
	GPODE-shooting	$\mathbf{9.11 \pm 0.37}$	$\mathbf{8.38 \pm 1.23}$	$\mathbf{10.11 \pm 0.79}$	$\mathbf{11.66 \pm 0.73}$	26.72 ± 0.63	$\mathbf{21.17 \pm 2.88}$

plications where uncertainty quantification is critical. Many conventional machine learning algorithms have been interpreted and modeled as continuous-time dynamical systems, with applications to generative modeling (Grathwohl et al., 2019) and probabilistic alignment (Ustyuzhaninov et al., 2020), among others. However, scaling GPs to high-dimensional datasets (such as images) can be a bottleneck. The applicability of the proposed model as a plug-in extension for these applications can be studied as part of future work.

We also highlighted a problem of learning black-box ODE models on long trajectories and proposed a probabilistic shooting framework enabling efficient inference on such tasks. This framework can be applied to other existing approaches, such as NeuralODEs. However, the proposed shooting augmentation introduces model approximation and involves approximating inference over auxiliary shooting variables. Hence the benefits of the shooting augmentation can be task specific, especially on short sequences. Comprehensive empirical studies across different types of tasks can be considered in future work.

References

Tarmo Äijö and Harri Lähdesmäki. Learning gene regulatory networks from gene expression measurements using non-parametric molecular kinetics. *Bioinformatics*, 25 (22):2937–2944, 2009.

Cedric Archambeau, Dan Cornford, Manfred Opper, and John Shawe-Taylor. Gaussian process approximations of stochastic differential equations. In *Gaussian Processes in Practice*, pages 1–16. PMLR, 2007.

Hans Bock and Karl Plitt. A multiple shooting algorithm for direct solution of optimal control problems. In *IFAC World Congress*, pages 242–247, 1984.

Hans Georg Bock. Recent advances in parameter identification techniques for ode. *Numerical treatment of inverse problems in differential and integral equations*, pages 95–121, 1983.

John Butcher and Nicolette Goodwin. *Numerical methods*

for ordinary differential equations. Wiley Online Library, 2nd edition, 2008.

Ben Calderhead, Mark Girolami, and Neil Lawrence. Accelerating Bayesian inference over nonlinear differential equations with Gaussian processes. In *Advances in Neural Information Processing Systems*, 2008.

Tian Qi Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. Neural ordinary differential equations. In *Advances in Neural Information Processing Systems*, pages 6571–6583, 2018.

Krzysztof Choromanski, Jared Davis, Valerii Likhosherstov, Xingyou Song, Jean-Jacques Slotine, Jacob Varley, Honglak Lee, Adrian Weller, and Vikas Sindhwani. An Ode to an ODE. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

Raj Dandekar, Karen Chung, Vaibhav Dixit, Mohamed Tarek, Aslan Garcia-Valadez, Krishna Vishal Vemula, and Chris Rackauckas. Bayesian neural ordinary differential equations. *arXiv preprint arXiv:2012.07244*, 2020.

Moritz Diehl and Sebastian Gros. *Numerical Optimal Control*. University of Freiburg, 2017.

Andreas Doerr, Christian Daniel, Martin Schiegg, Nguyen-Tuong Duy, Stefan Schaal, Marc Toussaint, and Trimpe Sebastian. Probabilistic recurrent state-space models. In *International Conference on Machine Learning*, pages 1280–1289, 2018.

Frank Dondelinger, Dirk Husmeier, Simon Rogers, and Maurizio Filippone. ODE parameter inference using adaptive gradient matching with Gaussian processes. In *Artificial Intelligence and Statistics*, pages 216–228, 2013.

Lea Duncker, Gergo Bohner, Julien Boussard, and Maneesh Sahani. Learning interpretable continuous-time models of latent stochastic dynamical systems. In *International Conference on Machine Learning*, pages 1726–1734. PMLR, 2019.

Stefanos Eleftheriadis, Tom Nicholson, Marc Peter Deisenroth, and James Hensman. Identification of Gaussian process state space models. In *Advances in Neural Information Processing Systems*, pages 5309–5319, 2017.

Katharina Ensinger, Friedrich Solowjow, Michael Tiemann, and Sebastian Trimpe. Symplectic gaussian process dynamics. *arXiv preprint arXiv:2102.01606*, 2021.

Roger Frigola, Yutian Chen, and Carl Rasmussen. Variational Gaussian process state-space models. In *Advances in Neural Information Processing Systems*, 2014.

Mark Girolami. Bayesian inference for differential equations. *Theoretical Computer Science*, 408:4–16, 2008.

Will Grathwohl, Ricky T. Q. Chen, Jesse Bettencourt, Ilya Sutskever, and David Duvenaud. FFJORD: Free-form continuous dynamics for scalable reversible generative models. In *ICLR*, 2019.

Eldad Haber and Lars Ruthotto. Stable architectures for deep neural networks. *Inverse Problems*, 34:014004, 2017.

Markus Heinonen and Florence d'Alché-Buc. Learning nonparametric differential equations with operator-valued kernels and gradient matching. Technical report, Universite d'Evry, 2014.

Markus Heinonen, Cagatay Yildiz, Henrik Mannerström, Jukka Intosalmi, and Harri Lähdesmäki. Learning unknown ODE models with Gaussian processes. In *International Conference on Machine Learning*, pages 1959–1968, 2018.

James Henderson and George Michailidis. Network reconstruction using nonparametric additive ODE models. *PLOS ONE*, 9(4):e94003, 2014.

James Hensman, Nicolò Fusi, and Neil Lawrence. Gaussian processes for big data. In *Uncertainty in Artificial Intelligence*, pages 282–290, 2013.

James Hensman, Alexander G Matthews, Maurizio Filippone, and Zoubin Ghahramani. Mcmc for variationally sparse gaussian processes. *Advances in Neural Information Processing Systems*, 28:1648–1656, 2015.

Morris Hirsch, Stephen Smale, and Robert Devaney. *Differential equations, dynamical systems, and an introduction to chaos*. Academic press, 2012.

Alessandro Davide Ialongo, Mark Van Der Wilk, James Hensman, and Carl Edward Rasmussen. Overcoming mean-field approximations in recurrent gaussian process models. In *International Conference on Machine Learning*, pages 2931–2940, 2019.

Martin Jørgensen, Marc Deisenroth, and Hugh Salimbeni. Stochastic differential equations with variational wishart diffusions. In *International Conference on Machine Learning*, pages 4974–4983. PMLR, 2020.

Diederik P Kingma and Jimmy Lei Ba. Adam: A method for stochastic optimization. In *Proceedings of 3rd International Conference on Learning Representations*, 2014.

Petar Kokotovic and James Heller. Direct and adjoint sensitivity equations for parameter optimization. *IEEE Transactions on Automatic Control*, 12(5):609–610, 1967.

Xuechen Li, Ting-Kam Leonard Wong, Ricky TQ Chen, and David Duvenaud. Scalable gradients for stochastic differential equations. In *International Conference on Artificial Intelligence and Statistics*, pages 3870–3882. PMLR, 2020.

Ernst Lindelöf. Sur l'application de la méthode des approximations successives aux équations différentielles ordinaires du premier ordre. *Comptes rendus hebdomadaires des séances de l'Académie des sciences*, 116:454–457, 1894.

Stefano Massaroli, Michael Poli, Sho Sonoda, Taiji Suzuki, Jinkyoo Park, Atsushi Yamashita, and Hajime Asama. Differentiable multiple shooting layers. *Advances in Neural Information Processing Systems*, 34, 2021.

Olga Mikheeva, Ieva Kazlauskaite, Adam Hartshorne, Hedvig Kjellström, Carl Henrik Ek, and Neill DF Campbell. Aligned multi-task gaussian process. *arXiv preprint arXiv:2110.15761*, 2021.

Michael Osborne. On shooting methods for boundary value problems. *Journal of Mathematical Analysis and Applications*, 27:417–433, 1969.

Lev Pontryagin, Evgenii Mishchenko, Vladimir Boltyanskii, and Revas Gamkrelidze. *The mathematical theory of optimal processes*. Interscience Publishers, 1962. Translation KN Trirogoff.

Christopher Rackauckas and Qing Nie. Differentialequations.jl – a performant and feature-rich ecosystem for solving differential equations in Julia. *Journal of Open Research Software*, 5(1), 2017.

Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *Advances in Neural Information Processing Systems*, pages 1177–1184, 2007.

Jim Ramsay, Giles Hooker, David Campbell, and Jiguo Cao. Parameter estimation for differential equations: A generalized smoothing approach. *Journal of the Royal Statistical Society: Series B*, 69(5):741–796, 2007.

Carl Rasmussen and Christopher Williams. *Gaussian processes for machine learning*. The MIT Press, 2006.

Yulia Rubanova, Ricky T. Q. Chen, and David Duvenaud. Latent ODEs for irregularly-sampled time series. In *Advances in Neural Information Processing Systems*, 2019.

Development Team Stan. Stan modeling language users guide and reference manual, mc-stan.org. 2021.

Michalis Titsias. Variational learning of inducing variables in sparse Gaussian processes. In *Artificial Intelligence and Statistics*, pages 567–574, 2009.

Evren Mert Turan and Johannes Jäschke. Multiple shooting with neural differential equations. *arXiv preprint arXiv:2109.06786*, 2021.

Ryan Turner, Marc Deisenroth, and Carl Rasmussen. State-space inference and learning with Gaussian processes. In *Artificial Intelligence and Statistics*, pages 868–875, 2010.

Belinda Tzen and Maxim Raginsky. Neural stochastic differential equations: Deep latent gaussian models in the diffusion limit. *arXiv preprint arXiv:1905.09883*, 2019.

Ivan Ustyuzhaninov, Ieva Kazlauskaite, Carl Henrik Ek, and Neill Campbell. Monotonic Gaussian process flows. In *Artificial Intelligence and Statistics*, volume 108, pages 3057–3067, 2020.

B vanDomselaar and Piet W Hemker. Nonlinear parameter estimation in initial value problems. *Stichting Mathematisch Centrum. Numerieke Wiskunde*, (NW 18/75), 1975.

James Varah. A spline least squares method for numerical parameter estimation in differential equations. *SIAM Journal on Scientific and Statistical Computing*, 3(1):28–46, 1982.

Jack Wang, Aaron Hertzmann, and David Fleet. Gaussian process dynamical models. In *Advances in Neural Information Processing Systems*, 2005.

Jack Wang, David Fleet, and Aaron Hertzmann. Gaussian process dynamical models for human motion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(2):283–298, 2008.

Philippe Wenk, Gabriele Abbati, Michael Osborne, Bernhard Schölkopf, Andreas Krause, and Stefan Bauer. ODIN: ODE-informed regression for parameter and state inference in time-continuous dynamical systems. In *AAAI Conference on Artificial Intelligence*, volume 34, pages 6364–6371, 2020.

James Wilson, Viacheslav Borovitskiy, Alexander Terenin, Peter Mostowsky, and Marc Deisenroth. Efficiently sampling functions from Gaussian process posteriors. In *International Conference on Machine Learning*, pages 10292–10302, 2020.

Cagatay Yildiz, Markus Heinonen, and Harri Lähdesmäki. ODE2VAE: Deep generative second order ODEs with Bayesian neural networks. In *Advances in Neural Information Processing Systems*, pages 13412–13421, 2019.

Juntang Zhuang, Nicha Dvornek, Xiaoxiao Li, Sekhar Tatikonda, Xenophon Papademetris, and James Duncan. Adaptive checkpoint adjoint method for gradient estimation in neural ODE. In *International Conference on Machine Learning*, pages 11639–11649, 2020.

Juntang Zhuang, Nicha Dvornek, Sekhar Tatikonda, and James Duncan. MALI: A memory efficient and reverse accurate integrator for neural ODEs. In *ICLR*, 2021.

Appendix : Variational multiple shooting for Bayesian ODEs with Gaussian processes

S1 DETAILED DERIVATIONS

S1.1 INFERENCE FOR THE VANILLA GPODE MODEL

The model. We consider the problem of inferring an ODE system

$$\mathbf{y}(t) = \mathbf{x}(t) + \boldsymbol{\epsilon} \tag{1}$$

$$\mathbf{x}(t) = \mathbf{x}_0 + \int_0^t \mathbf{f}(\mathbf{x}(\tau))d\tau \tag{2}$$

from some noisy observations $\mathbf{y}(t)$ of the true system state $\mathbf{x}(t) \in \mathbb{R}^D$, whose evolution over time $t \in \mathbb{R}_+$ follows a differential equation vector field

$$\dot{\mathbf{x}}(t) = \frac{d\mathbf{x}(t)}{dt} := \mathbf{f}(\mathbf{x}(t)), \qquad \mathbf{f} : \mathbb{R}^D \mapsto \mathbb{R}^D \tag{3}$$

starting from an initial state $\mathbf{x}_0 \in \mathbb{R}^D$. Our goal is to learn the underlying ODE vector field $\mathbf{f}$.

We propose a Gaussian process prior to the differential function

$$\mathbf{f}(\mathbf{x}) \sim \mathcal{GP}(\mathbf{0}, k(\mathbf{x}, \mathbf{x}')). \tag{4}$$

Following Titsias (2009) for sparse inference of GPs using inducing variables, we augment the full model with inducing values $\mathbf{U} = (\mathbf{u}_1, \ldots, \mathbf{u}_M)^T \in R^{M\times D}$ and inducing locations $\mathbf{Z} = (\mathbf{z}_1, \ldots, \mathbf{z}_M)^T \in R^{M\times D}$, which results in a low-rank GP

$$p(\mathbf{U}) = \mathcal{N}(\mathbf{U}|\mathbf{0}, \mathbf{K}_{\mathbf{ZZ}}) \tag{5}$$

$$p(\mathbf{f}|\mathbf{U}) = \mathcal{N}(\mathbf{f}|\mathbf{A}\mathrm{vec}(\mathbf{U}), \mathbf{K}_{\mathbf{XX}} - \mathbf{A}\mathbf{K}_{\mathbf{ZZ}}\mathbf{A}^T), \tag{6}$$

where $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \ldots \mathbf{x}_{N'})^T \in \mathbb{R}^{N'\times D}$ collects all the intermediate state evaluations $\mathbf{x}(t_i)$ encountered along numerical approximation of the true continuous ODE integral (2), $\mathbf{f} = (\mathbf{f}(\mathbf{x}_1)^T, \ldots, \mathbf{f}(\mathbf{x}_{N'})^T)^T \in \mathbb{R}^{N'D\times 1}$, $\mathbf{K}_{\mathbf{XX}}$ is a block-partitioned matrix of size $N'D \times N'D$ with $D \times D$ blocks, so that block $(\mathbf{K}_{\mathbf{XX}})_{i,j} = K(\mathbf{x}_i, \mathbf{x}_j)$, and $\mathbf{A} = \mathbf{K}_{\mathbf{XZ}}\mathbf{K}_{\mathbf{ZZ}}^{-1}$.

The joint model The joint probability of the model is

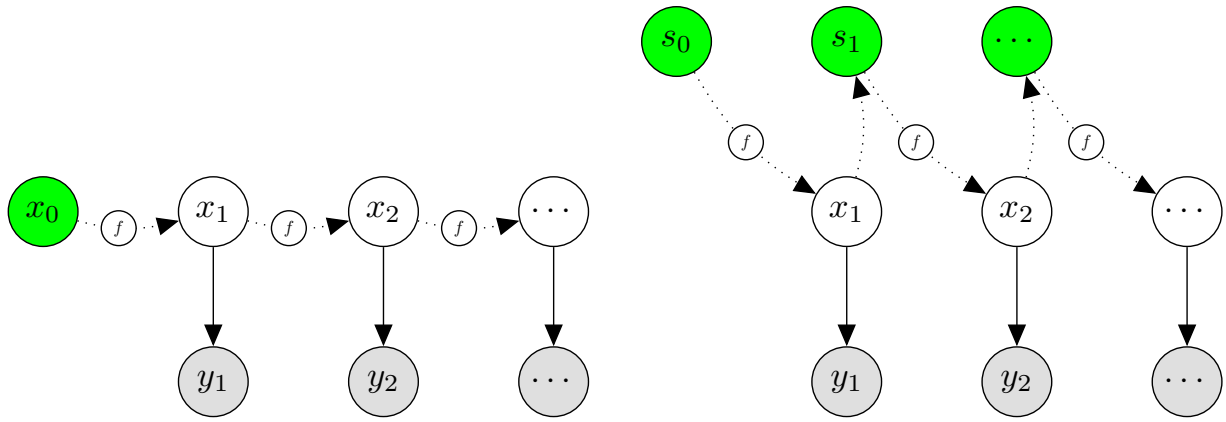
$$p(\mathbf{Y}, \mathbf{f}, \mathbf{U}, \mathbf{x}_0) = \prod_{i=1}^N p(\mathbf{y}_i|\mathbf{f}, \mathbf{x}_0)p(\mathbf{f}|\mathbf{U})p(\mathbf{U})p(\mathbf{x}_0) \tag{7}$$

$$= \prod_{i=1}^N \underbrace{p(\mathbf{y}_i|\mathbf{f}, \mathbf{x}_0)}_{\text{likelihood}} \underbrace{p(\mathbf{f}, \mathbf{U})}_{\text{GP prior}} \underbrace{p(\mathbf{x}_0)}_{\text{initial state prior}}, \tag{8}$$

where we assume a standard Gaussian prior $p(\mathbf{x}_0) = \mathcal{N}(\mathbf{0}, \mathbf{I})$ for the unknown initial state $\mathbf{x}_0$.

Inference. Our primary goal is to learn the vector field $\mathbf{f}$ by inferring the model posterior $p(\mathbf{f}, \mathbf{U}, \mathbf{x}_0|\mathbf{Y})$, which is intractable. We resort to stochastic variational inference Hensman et al. (2013), and introduce a factorized Gaussian posterior approximation for the inducing variables across state dimensions

$$q(\mathbf{U}) = \prod_{d=1}^D \mathcal{N}(\mathbf{u}_d|\mathbf{m}_d, \mathbf{Q}_d), \tag{9}$$



(a) Vanilla GPODE model

(b) Shooting GPODE model

Figure 1: Plate diagrams: latent random variables that are considered during model inference are shaded in green. The intermediate variables $\mathbf{x}_i$ (unshaded) are defined as deterministic transformations of the inferred variables (conditioned on the vectorfield). In the vanilla GPODE formulation (a), the initial state distribution $\mathbf{x}_0$ is integrated forward in time to match all the observations $\{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_N\}$ forming a full trajectory. The shooting version (b) splits the full trajectory into multiple subintervals. Every subinterval i starts with an approximated state distribution $\mathbf{s}_i$, which is integrated forward to match the next observation $\mathbf{y}_{i+1}$. In addition, the state evolution from the previous shooting variable is matched to the variational shooting approximation at the current state.

where, $\mathbf{u}_d \in \mathbb{R}^M$ and $\mathbf{m}_d \in \mathbb{R}^M, \mathbf{Q}_d \in \mathbb{R}^{M \times M}$ are the mean and covariance parameters of the variational Gaussian posterior approximation for the inducing variables. The Gaussian process posterior process with an inducing approximation can be written as

$$q(\mathbf{f}) = \int p(\mathbf{f}|\mathbf{U})q(\mathbf{U})d\mathbf{U} \tag{10}$$

$$= \int \mathcal{N}\left(\mathbf{f}|\mathbf{A}\text{vec}(\mathbf{U}), \mathbf{K}_{\mathbf{XX}} - \mathbf{A}\mathbf{K}_{\mathbf{ZZ}}\mathbf{A}^T\right) q(\mathbf{U})d\mathbf{U}. \tag{11}$$

We also introduce posterior approximation for the initial state variable $\mathbf{x}_0$,

$$q(\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_0|\mathbf{m}_0, \mathbf{S}_0). \tag{12}$$

This results in a variational joint posterior approximation

$$q(\mathbf{f}, \mathbf{U}, \mathbf{x}_0) = q(\mathbf{f}, \mathbf{U})q(\mathbf{x}_0) \tag{13}$$

$$= p(\mathbf{f}|\mathbf{U})q(\mathbf{U})q(\mathbf{x}_0), \tag{14}$$

ELBO. With the above model specification, under variational inference of the posterior approximations, the evidence lower bound (ELBO) $\log p(\mathbf{Y}) \geq \mathcal{L}$ can be written as,

$$\mathcal{L} = \iiint q(\mathbf{f}, \mathbf{U}, \mathbf{x}_0) \log \frac{p(\mathbf{Y}, \mathbf{f}, \mathbf{U}, \mathbf{x}_0)}{q(\mathbf{f}, \mathbf{U}, \mathbf{x}_0)} d\mathbf{f}d\mathbf{U}d\mathbf{x}_0 \tag{15}$$

$$= \iiint q(\mathbf{f}, \mathbf{U}, \mathbf{x}_0) \log \prod_{i=1}^{N} \underbrace{p(\mathbf{y}_i|\mathbf{f}, \mathbf{x}_0)}_{\mathcal{L}_y} \frac{p(\mathbf{f}|\mathbf{U})}{p(\mathbf{f}|\mathbf{U})} \underbrace{\frac{p(\mathbf{U})}{q(\mathbf{U})}}_{\mathcal{L}_u} \underbrace{\frac{p(\mathbf{x}_0)}{q(\mathbf{x}_0)}}_{\mathcal{L}_{\mathbf{x}_0}} d\mathbf{f}d\mathbf{U}d\mathbf{x}_0. \tag{16}$$

Hence the ELBO decomposes into three additive terms

$$\mathcal{L} = \mathcal{L}_y + \mathcal{L}_u + \mathcal{L}_{\mathbf{x}_0}, \tag{17}$$

where each term contains the (relevant parts of) expectation over $q(\mathbf{f}, \mathbf{U}, \mathbf{x}_0)$.

Likelihood term. The variational likelihood term $\mathcal{L}_y$ is an expectation of the likelihood wrt the variationally marginalized vectorfield posterior $q(\mathbf{f})$, and the initial state distribution $q(\mathbf{x}_0)$,

$$\mathcal{L}_y = \iint q(\mathbf{f}, \mathbf{x}_0) \log p(\mathbf{y}|\mathbf{f}, \mathbf{x}_0) d\mathbf{f} d\mathbf{x}_0 \tag{18}$$

$$= \sum_{i=1}^{N} \mathbb{E}_{q(\mathbf{f},\mathbf{x}_0)} \log p(\mathbf{y}_i|\mathbf{f}, \mathbf{x}_0). \tag{19}$$

This term computes the likelihood $p(\mathbf{y}_i|\mathbf{f}, \mathbf{x}_0) = p(\mathbf{y}_i|\mathbf{x}_i)$ over ODE state solutions $\mathbf{x}_i = \mathbf{x}_0 + \int_0^{t_i} \mathbf{f}(\mathbf{x}(\tau))d\tau$ for a single realization of the vector field $\mathbf{f} \sim p(\mathbf{f})$ and the initial state $\mathbf{x}_0 \sim p(\mathbf{x}_0)$. Because of the non-linear integration $\mathbf{x}_0 \mapsto \mathbf{x}(t)$, we cannot solve this integral analytically. Instead, we resort to Monte Carlo integration by sampling ODE trajectories over different vector field realizations $\mathbf{f} \sim q(\mathbf{f})$ and initial states $\mathbf{x}_0 \sim q(\mathbf{x}_0)$. In practice, this term can be approximated as

$$\mathcal{L}_y \approx \frac{1}{S} \sum_{s=1}^{S} \sum_{i=1}^{N} \log p(\mathbf{y}_i|\mathbf{f}^{(s)}, \mathbf{x}_0^{(s)}) \tag{20}$$

where we sum over S reparameterized samples $\mathbf{f}^{(s)} \sim q(\mathbf{f})$ and $\mathbf{x}_0^{(s)} \sim q(\mathbf{x}_0)$.

Inducing KL. This term corresponds to the KL divergence between variational posterior and the prior distribution of inducing values. This term can be derived analytically as the KL between multivariate Gaussians.

$$\mathcal{L}_u = \int q(\mathbf{U}) \log \frac{p(\mathbf{U})}{q(\mathbf{U})} d\mathbf{U} \tag{21}$$

$$= \sum_{d=1}^{D} \int q(\mathbf{u}_d) \log \frac{p(\mathbf{u}_d)}{q(\mathbf{u}_d)} d\mathbf{u} \tag{22}$$

$$= -\sum_{d=1}^{D} \mathrm{KL}\left[q(\mathbf{u}_d)||p(\mathbf{u}_d)\right] \tag{23}$$

Initial state KL. This term corresponds to the KL divergence between variational posterior and the prior distribution of the initial state. With an assumption of Gaussian prior and variational posterior, this term can also be derived analytically,

$$\mathcal{L}_{\mathbf{x}_0} = \int q(\mathbf{x}_0) \log \frac{p(\mathbf{x}_0)}{q(\mathbf{x}_0)} d\mathbf{x}_0 \tag{24}$$

$$= -\mathrm{KL}\left[q(\mathbf{x}_0)||p(\mathbf{x}_0)\right] \tag{25}$$

Complete ELBO. The full ELBO is then

$$\mathcal{L} = \sum_{i=1}^{N} \mathbb{E}_{q(\mathbf{f},\mathbf{x}_0)} \log p(\mathbf{y}_i|\mathbf{f}, \mathbf{x}_0) - \mathrm{KL}[q(\mathbf{U}) \,||\, p(\mathbf{U})] - \mathrm{KL}[q(\mathbf{x}_0) \,||\, p(\mathbf{x}_0)] \tag{26}$$

S1.2 DECOUPLED SAMPLING OF GPODES

In this section, we provide details for simulating valid ODE trajectories from a GP vector field posterior of the form

$$q(\mathbf{u}) = \mathcal{N}(\mathbf{m}, \mathbf{Q}), \tag{27}$$

$$q(\mathbf{f}) = \int p(\mathbf{f}|\mathbf{u}) q(\mathbf{u}) d\mathbf{u} \tag{28}$$

$$= \int \mathcal{N}\left(\mathbf{f}|\mathbf{A}\mathbf{u}, \mathbf{K}_{\mathbf{XX}} - \mathbf{A}\mathbf{K}_{\mathbf{ZZ}}\mathbf{A}^T\right) q(\mathbf{u}) d\mathbf{u}, \tag{29}$$

where $\mathbf{A} = \mathbf{K}_{\mathbf{XZ}}\mathbf{K}_{\mathbf{ZZ}}^{-1}$ and $\mathbf{m} \in \mathbb{R}^M, \mathbf{Q} \in \mathbb{R}^{M \times M}$ are the variational mean and covariance parameters of the Gaussian posterior approximation for inducing variables. For simplicity, we consider a scalar valued GP, but it is straightforward to extend this approach to vector-valued GPs.

A sparse GP posterior of the form (29) can be decomposed into two parts using Matheron's rule (Corollary 2 Wilson et al. (2020)),

$$\underbrace{f(\mathbf{x})|\mathbf{u}}_{\texttt{posterior}} = \underbrace{f(\mathbf{x})}_{\texttt{prior}} + \underbrace{k(\mathbf{x},\mathbf{Z})K(\mathbf{Z},\mathbf{Z})^{-1}(\mathbf{u}-\mathbf{f}_{\mathbf{Z}})}_{\texttt{update}}. \tag{30}$$

Wilson et al. (2020) propose a decoupled sampling from the `posterior` by using different bases for the `prior` and `update` terms. In particular, they propose Fourier basis functions for the `prior` term and canonical basis for the `update` term respectively

$$\underbrace{f(\mathbf{x})|\mathbf{u}}_{\texttt{posterior}} \approx \underbrace{\sum_{i=1}^{S} w_i\phi_i(\mathbf{x})}_{\texttt{prior}} + \underbrace{\sum_{j=1}^{M} \nu_j K(\mathbf{x},\mathbf{z}_j)}_{\texttt{update}}, \tag{31}$$

where we use S Fourier bases $\phi_i(\cdot)$ with $w_i \sim \mathcal{N}(0,1)$ (Rahimi and Recht, 2007) to represent the stationary prior, and function basis $K(\cdot,\mathbf{z}_j)$ for the posterior update with $\boldsymbol{\nu} = K(\mathbf{Z},\mathbf{Z})^{-1}(\mathbf{u}-\boldsymbol{\Phi}\mathbf{w})$, $\boldsymbol{\Phi} = \boldsymbol{\phi}(\mathbf{Z}) \in \mathbb{R}^{M\times S}$, $\mathbf{w} \in \mathbb{R}^S$. We can evaluate functions from the posterior (29) in linear time at arbitrary locations.

For the experimental results presented in the paper, we use a squared exponential kernel for which we can compute the feature maps $\boldsymbol{\phi}_i(\mathbf{x}) = \sqrt{\frac{\sigma_f^2}{S}}(\cos \mathbf{x}^T\boldsymbol{\omega}_i, \sin \mathbf{x}^T\boldsymbol{\omega}_i)$ where $\boldsymbol{\omega}_i$ is sampled proportional to the spectral density of the squared exponential kernel $\boldsymbol{\omega}_i \sim \mathcal{N}(\mathbf{0},\Lambda^{-1})$, Λ is a diagonal matrix collecting lengthscale parameters of the kernel $\Lambda = \text{diag}(l_1^2, l_2^2, \ldots, l_D^2)$ and σ_f^2 is the signal variance parameter. In the case of the squared exponential kernel, this results in $2S$ feature maps $\boldsymbol{\phi}(\mathbf{x}) \in \mathbb{R}^{2S}$, for which we sample weights $\mathbf{w} \in \mathbb{R}^{2S}$ from the standard Normal $w_i \sim \mathcal{N}(0,1)$. By fixing random samples of feature maps $\boldsymbol{\phi}(\cdot)$, corresponding weights $\mathbf{w}$ and inducing values $\mathbf{u}$ for an ODE integration call, we can sample a unique ODE trajectory from a posterior vector field of the form (29).

S1.3 PROBABILISTIC SHOOTING FORMULATION FOR GPODES

The model. We consider the problem of inferring an ODE system

$$\mathbf{y}(t) = \mathbf{x}(t) + \boldsymbol{\epsilon} \tag{32}$$

$$\mathbf{x}(t) = \mathbf{s}_0 + \int_0^t \mathbf{f}(\mathbf{x}(\tau))d\tau, \tag{33}$$

from some noisy observations $\mathbf{y}(t)$ of the true system state $\mathbf{x}(t) \in \mathbb{R}^D$, whose evolution over time $t \in \mathbb{R}_+$ follows a differential equation

$$\dot{\mathbf{x}}(t) = \frac{d\mathbf{x}(t)}{dt} := \mathbf{f}(\mathbf{x}(t)), \qquad \mathbf{f} : \mathbb{R}^D \mapsto \mathbb{R}^D \tag{34}$$

starting from the initial state $\mathbf{s}_0 \in \mathbb{R}^D$. Our goal is to learn the underlying ODE vector field $\mathbf{f}$.

Shooting augmentation. We propose an augmented 'shooting' ODE system

$$\mathbf{y}_i = \mathbf{x}(t_i;\mathbf{s}_{i-1}) + \boldsymbol{\epsilon} \tag{35}$$

$$\mathbf{x}(t_i;\mathbf{s}_{i-1}) = \mathbf{s}_{i-1} + \int_{t_{i-1}}^{t_i} \mathbf{f}(\mathbf{x}(\tau))d\tau \tag{36}$$

$$\mathbf{s}_i = \mathbf{x}(t_i;\mathbf{s}_{i-1}) + \boldsymbol{\xi}, \tag{37}$$

where we divide the state function $\mathbf{x}(t)$ into N short segments, with the end state of ith segment $\mathbf{x}(t_i;\mathbf{s}_{i-1})$ defining solutions to initial value problems (36) starting from the corresponding shooting variables $\mathbf{s}_{i-1}$. These short shooting segments follow the same differential $\mathbf{f}$ as the original model. The augmented system is equivalent to the original ODE system, in the limit when the tolerance parameter $\boldsymbol{\xi} \to \mathbf{0}$.

We assume Gaussian distributions on both observation noise and tolerance parameters, resulting in the following distributions,

$$p(\mathbf{y}_i|\mathbf{s}_{i-1}) = \mathcal{N}(\mathbf{y}_i|\mathbf{x}(t_i;\mathbf{s}_{i-1}), \sigma_y^2\mathbf{I}); \qquad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \sigma_y^2\mathbf{I}), \tag{38}$$

$$p(\mathbf{s}_i|\mathbf{s}_{i-1}) = \mathcal{N}(\mathbf{s}_i|\mathbf{x}(t_i;\mathbf{s}_{i-1}), \sigma_\xi^2\mathbf{I}); \qquad \boldsymbol{\xi} \sim \mathcal{N}(\mathbf{0}, \sigma_\xi^2\mathbf{I}). \tag{39}$$

Gaussian process ODE. We propose a Gaussian process prior for the differential function

$$\mathbf{f}(\mathbf{x}) \sim \mathcal{GP}(\mathbf{0}, k(\mathbf{x}, \mathbf{x}')) \tag{40}$$

In addition, we augment the full model with inducing values $\mathbf{U} = (\mathbf{u}_1, \ldots, \mathbf{u}_M)^T \in R^{M \times D}$ and inducing locations $\mathbf{Z} = (\mathbf{z}_1, \ldots, \mathbf{z}_M)^T \in R^{M \times D}$, which results in a low-rank GP

$$p(\mathbf{U}) = \mathcal{N}(\mathbf{U}|\mathbf{0}, \mathbf{K}_{\mathbf{ZZ}}) \tag{41}$$

$$p(\mathbf{f}|\mathbf{U}) = \mathcal{N}(\mathbf{f}|\mathbf{A}\mathrm{vec}(\mathbf{U}), \mathbf{K}_{\mathbf{XX}} - \mathbf{A}\mathbf{K}_{\mathbf{ZZ}}\mathbf{A}^T), \tag{42}$$

where $\mathbf{A} = \mathbf{K}_{\mathbf{XZ}}\mathbf{K}_{\mathbf{ZZ}}^{-1}$.

The joint model. The joint probability of the model is

$$p(\mathbf{Y}, \mathbf{S}, \mathbf{f}, \mathbf{U}) = p(\mathbf{Y}|\mathbf{S}, \mathbf{f})p(\mathbf{S}|\mathbf{f})p(\mathbf{f}|\mathbf{U})p(\mathbf{U}) \tag{43}$$

$$= \prod_{i=1}^{N} \underbrace{p(\mathbf{y}_i|\mathbf{s}_{i-1}, \mathbf{f})}_{\text{likelihood}} \prod_{i=1}^{N-1} \underbrace{p(\mathbf{s}_i|\mathbf{s}_{i-1}, \mathbf{f})}_{\text{shooting prior}} \underbrace{p(\mathbf{s}_0)}_{\text{initial state}} \underbrace{p(\mathbf{f}|\mathbf{U})p(\mathbf{U})}_{\text{GP prior}}, \tag{44}$$

where $\mathbf{S} = (\mathbf{s}_0, \mathbf{s}_1, \ldots \mathbf{s}_{N-1})^T \in \mathbb{R}^{N \times D}$ collects all shooting variables.

We also note that observations are at indices $1, \ldots, N$, while the shooting variables are always one behind the observations at $0, \ldots, N-1$ (see plate diagram 1 (b)).

Inference. Our primary goal is to learn the vector field $\mathbf{f}$ by inferring the model posterior $p(\mathbf{S}, \mathbf{f}, \mathbf{U}|\mathbf{Y})$, which is intractable. Similar to non-shooting GPODEs, we introduce a factorized Gaussian posterior approximation for the inducing variables across state dimensions

$$q(\mathbf{U}) = \prod_{d=1}^{D} \mathcal{N}(\mathbf{u}_d|\mathbf{m}_d, \mathbf{Q}_d), \tag{45}$$

where, $\mathbf{u}_d \in \mathbb{R}^M$ and $\mathbf{m}_d \in \mathbb{R}^M, \mathbf{Q}_d \in \mathbb{R}^{M \times M}$ are the mean and covariance parameters of the variational Gaussian posterior approximation for the inducing variables.

The Gaussian process posterior process with an inducing approximation can be written as

$$q(\mathbf{f}) = \int p(\mathbf{f}|\mathbf{U})q(\mathbf{U})d\mathbf{U} \tag{46}$$

$$= \int \mathcal{N}\left(\mathbf{f}|\mathbf{A}\mathrm{vec}(\mathbf{U}), \mathbf{K}_{\mathbf{XX}} - \mathbf{A}\mathbf{K}_{\mathbf{ZZ}}\mathbf{A}^T\right) q(\mathbf{U})d\mathbf{U}. \tag{47}$$

Next, we introduce a factorized Gaussian posterior approximations for the shooting variables $\mathbf{S}$ as well,

$$q(\mathbf{S}) = \prod_{i=0}^{N-1} q(\mathbf{s}_i) = \prod_{i=0}^{N-1} \mathcal{N}(\mathbf{s}_i|\mathbf{a}_i, \Sigma_i). \tag{48}$$

where, $\mathbf{a}_i \in \mathbb{R}^D$ and $\Sigma_i \in \mathbb{R}^{D \times D}$ are the mean and covariance parameters of the variational Gaussian posterior approximation for the shooting variables.

This results in a variational joint posterior approximation

$$q(\mathbf{S}, \mathbf{f}, \mathbf{U}) = q(\mathbf{S})q(\mathbf{f}, \mathbf{U}) \tag{49}$$

$$= \prod_{i=0}^{N-1} q(\mathbf{s}_i)p(\mathbf{f}|\mathbf{U})q(\mathbf{U}). \tag{50}$$

ELBO. Under variational inference the posterior approximations q are optimized to match the true posterior in the KL sense,

$$\arg\min_q \ \mathrm{KL}\left[q(\mathbf{S},\mathbf{f},\mathbf{U})\,||\,p(\mathbf{S},\mathbf{f},\mathbf{U}|\mathbf{Y})\right]. \tag{51}$$

This is equivalent to maximizing the evidence lower bound (ELBO) $\log p(\mathbf{Y}) \geq \mathcal{L}$,

$$\mathcal{L} = \iiint q(\mathbf{S},\mathbf{f},\mathbf{U}) \log\left[\frac{p(\mathbf{Y},\mathbf{S},\mathbf{f},\mathbf{U})}{q(\mathbf{S},\mathbf{f},\mathbf{U})}\right] d\mathbf{S}d\mathbf{f}d\mathbf{U} \tag{52}$$

$$= \iiint q(\mathbf{S},\mathbf{f},\mathbf{U}) \log\left[\prod_{i=1}^{N} p(\mathbf{y}_i|\mathbf{s}_{i-1},\mathbf{f}) \cdot \prod_{i=1}^{N-1} \frac{p(\mathbf{s}_i|\mathbf{s}_{i-1},\mathbf{f})}{q(\mathbf{s}_i)} \cdot \frac{p(\mathbf{s}_0)}{q(\mathbf{s}_0)} \cdot \frac{p(\mathbf{f},\mathbf{U})}{q(\mathbf{f},\mathbf{U})}\right] d\mathbf{S}d\mathbf{f}d\mathbf{U} \tag{53}$$

$$= \underbrace{\iint q(\mathbf{S})q(\mathbf{f}) \log \prod_{i=1}^{N} p(\mathbf{y}_i|\mathbf{s}_{i-1},\mathbf{f}) d\mathbf{S}d\mathbf{f}}_{\mathcal{L}_y} + \underbrace{\iint q(\mathbf{S})q(\mathbf{f}) \log \prod_{i=1}^{N-1} p(\mathbf{s}_i|\mathbf{s}_{i-1},\mathbf{f}) d\mathbf{S}d\mathbf{f}}_{\mathcal{L}_{sc}}$$

$$\underbrace{-\int q(\mathbf{S}) \log \prod_{i=1}^{N-1} q(\mathbf{s}_i) d\mathbf{S}}_{\mathcal{L}_{se}} + \underbrace{\int q(\mathbf{s}_0) \log \frac{p(\mathbf{s}_0)}{q(\mathbf{s}_0)} d\mathbf{s}_0}_{\mathcal{L}_0} + \underbrace{\int q(\mathbf{U}) \log \frac{p(\mathbf{U})}{q(\mathbf{U})} d\mathbf{U}}_{\mathcal{L}_u} \tag{54}$$

$$\tag{55}$$

which results in the ELBO decomposing into four additive terms

$$\mathcal{L} = \mathcal{L}_y + \mathcal{L}_{sc} + \mathcal{L}_{se} + \mathcal{L}_0 + \mathcal{L}_u, \tag{56}$$

where each term contains the (relevant parts of) expectation over $q(\mathbf{S},\mathbf{f},\mathbf{U})$.

Likelihood term. The variational likelihood term $\mathcal{L}_y$ is an expectation of the likelihood under the posteriors of shooting variables $q(\mathbf{S})$ and the posterior vectorfield $q(\mathbf{f})$,

$$\mathcal{L}_y = \iint q(\mathbf{S})q(\mathbf{f}) \log \prod_{i=1}^{N} p(\mathbf{y}_i|\mathbf{s}_{i-1},\mathbf{f}) d\mathbf{S}d\mathbf{f} \tag{57}$$

$$= \sum_{i=1}^{N} \iint q(\mathbf{s}_{i-1})q(\mathbf{f}) \log p(\mathbf{y}_i|\mathbf{s}_{i-1},\mathbf{f}) d\mathbf{s}_{i-1}d\mathbf{f} \tag{58}$$

$$= \sum_{i=1}^{N} \mathbb{E}_{q(\mathbf{s}_{i-1})q(\mathbf{f})}\Big[\log p(\mathbf{y}_i|\mathbf{s}_{i-1},\mathbf{f})\Big]. \tag{59}$$

We can evaluate this term with Monte Carlo integration by taking reparameterized samples from the posteriors $\mathbf{f}^{(s)} \sim q(\mathbf{f})$ and $\mathbf{s}_{i-1}^{(s)} \sim q(\mathbf{s}_{i-1})$ as below

$$\mathcal{L}_y = \sum_{i=1}^{N} \mathbb{E}_{q(\mathbf{s}_{i-1},\mathbf{f})}\Big[\log p(\mathbf{y}_i|\mathbf{s}_{i-1},\mathbf{f})\Big] \tag{60}$$

$$\mathcal{L}_y \approx \frac{1}{S}\sum_{s=1}^{S}\sum_{i=1}^{N}\Big[\log p(\mathbf{y}_i|\mathbf{x}_i^{(s)})\Big], \tag{61}$$

where $\mathbf{x}_i^{(s)}$ is defined as solution to the following initial value problem,

$$\mathbf{x}_i^{(s)} := \mathbf{x}^{(s)}(t_i;\mathbf{s}_{i-1}) = \mathbf{s}_{i-1}^{(s)} + \int_{t_{i-1}}^{t_i} \mathbf{f}^{(s)}(\mathbf{x}(\tau))d\tau. \tag{62}$$

Shooting cross-entropy term. This term computes the cross-entropy between the prior specification for the shooting variables under the ODE evolution $p(\mathbf{s}_i|\mathbf{s}_{i-1}, \mathbf{f})$, and the point-wise approximations $q(\mathbf{s}_i)$,

$$\mathcal{L}_{se} = \iint q(\mathbf{S})q(\mathbf{f})\Big[\log \prod_{i=1}^{N-1} p(\mathbf{s}_i|\mathbf{s}_{i-1}, \mathbf{f})\Big] d\mathbf{S}d\mathbf{f} \tag{63}$$

$$= \iint q(\mathbf{s}_{N-1}) \cdots q(\mathbf{s}_1)q(\mathbf{s}_0)q(\mathbf{f})\Big[\log p(\mathbf{s}_{N-1}|\mathbf{s}_{N-2}, \mathbf{f}) \cdots p(\mathbf{s}_1|\mathbf{s}_0, \mathbf{f})\Big] d\mathbf{S}d\mathbf{f} \tag{64}$$

$$= \sum_{i=1}^{N-1} \iint q(\mathbf{f})q(\mathbf{s}_i)q(\mathbf{s}_{i-1})\Big[\log p(\mathbf{s}_i|\mathbf{s}_{i-1}, \mathbf{f})\Big] d\mathbf{s}_{i-1}d\mathbf{s}_i d\mathbf{f} \tag{65}$$

$$= \sum_{i=1}^{N-1} \mathbb{E}_{q(\mathbf{s}_i, \mathbf{s}_{i-1}, \mathbf{f})}\Big[\log p\left(\mathbf{s}_i|\mathbf{s}_{i-1}, \mathbf{f}\right)\Big]. \tag{66}$$

This term can also be numerically estimated with Monte Carlo integration using posterior samples $\mathbf{f}^{(s)} \sim q(\mathbf{f})$, $\mathbf{s}_{i-1}^{(s)} \sim q(\mathbf{s}_{i-1})$ and $\mathbf{s}_i^{(s)} \sim q(\mathbf{s}_i)$

$$\mathcal{L}_{se} = \sum_{i=1}^{N-1} \mathbb{E}_{q(\mathbf{s}_i, \mathbf{s}_{i-1}, \mathbf{f})}\Big[\log p\left(\mathbf{s}_i|\mathbf{s}_{i-1}, \mathbf{f}\right)\Big] \tag{67}$$

$$\approx \frac{1}{S} \sum_{s=1}^{S} \sum_{i=1}^{N-1} \log p\left(\mathbf{s}_i^{(s)}|\mathbf{x}_i^{(s)}\right), \tag{68}$$

$$\mathbf{x}_i^{(s)} := \mathbf{x}^{(s)}(t_i; \mathbf{s}_{i-1}) = \mathbf{s}_{i-1}^{(s)} + \int_{t_{i-1}}^{t_i} \mathbf{f}^{(s)}(\mathbf{x}(\tau))d\tau. \tag{69}$$

Shooting entropy term. This term computes the entropy of the posterior approximations for shooting variables $q(\mathbf{s}_i)$. Since we assume factorized Gaussian approximations, this term can be simplified analytically as the sum of Gaussian entropy.

$$\mathcal{L}_{se} = -\int q(\mathbf{S}) \log \prod_{i=1}^{N-1} q(\mathbf{s}_i) d\mathbf{S} \tag{70}$$

$$= -\sum_{i=1}^{N-1} \mathbb{E}_{q(\mathbf{s}_i)}\Big[\log q(\mathbf{s}_i)\Big]. \tag{71}$$

Initial state KL term. This term corresponds to the KL divergence between variational posterior and the prior distribution of the initial state. With the assumption of Gaussian prior and variational posterior, this term can also be derived analytically,

$$\mathcal{L}_0 = \int q(\mathbf{s}_0) \log \frac{p(\mathbf{s}_0)}{q(\mathbf{s}_0)} d\mathbf{s}_0 \tag{72}$$

$$= -\operatorname{KL}\left[q(\mathbf{s}_0)||p(\mathbf{s}_0)\right]. \tag{73}$$

Inducing KL term. This term corresponds to the KL divergence between variational posterior and prior distribution of inducing values. This term can also be derived analytically as the KL between multivariate Gaussians.

$$\mathcal{L}_u = \int q(\mathbf{U}) \log \frac{p(\mathbf{U})}{q(\mathbf{U})} d\mathbf{U} \tag{74}$$

$$= \sum_{d=1}^{D} \int q(\mathbf{u}_d) \log \frac{p(\mathbf{u}_d)}{q(\mathbf{u}_d)} d\mathbf{u} \tag{75}$$

$$= -\sum_{d=1}^{D} \operatorname{KL}\left[q(\mathbf{u}_d)||p(\mathbf{u}_d)\right]. \tag{76}$$

Complete ELBO. The full ELBO is then

$$\mathcal{L} = \mathcal{L}_y + \mathcal{L}_{sc} + \mathcal{L}_{se} + \mathcal{L}_0 + \mathcal{L}_u \tag{77}$$

$$= \sum_{i=1}^{N} \mathbb{E}_{q(\mathbf{s}_{i-1},\mathbf{f})}\Big[\log p(\mathbf{y}_i|\mathbf{s}_{i-1},\mathbf{f})\Big] + \sum_{i=1}^{N-1} \mathbb{E}_{q(\mathbf{s}_i,\mathbf{s}_{i-1},\mathbf{f})}\Big[\log p\left(\mathbf{s}_i|\mathbf{s}_{i-1},\mathbf{f}\right)\Big]$$

$$- \sum_{i=1}^{N-1} \mathbb{E}_{q(\mathbf{s}_i)}\Big[\log q(\mathbf{s}_i)\Big] - \mathrm{KL}[q(\mathbf{s}_0)\,||\,p(\mathbf{s}_0)] - \mathrm{KL}[q(\mathbf{U})\,||\,p(\mathbf{U})] \tag{78}$$

which in practice is numerically estimated with Monte Carlo integration

$$\mathcal{L} \approx \frac{1}{S}\sum_{s=1}^{S}\sum_{i=1}^{N}\Big[\log p(\mathbf{y}_i|\mathbf{x}_i^{(s)})\Big] + \frac{1}{S}\sum_{s=1}^{S}\sum_{i=1}^{N-1}\log p\left(\mathbf{s}_i^{(s)}|\mathbf{x}_i^{(s)}\right)$$

$$- \sum_{i=1}^{N-1} \mathbb{E}_{q(\mathbf{s}_i)}\Big[\log q(\mathbf{s}_i)\Big] - \mathrm{KL}[q(\mathbf{s}_0)\,||\,p(\mathbf{s}_0)] - \mathrm{KL}[q(\mathbf{U})\,||\,p(\mathbf{U})] \tag{79}$$

where $\mathbf{f}^{(s)} \sim q(\mathbf{f})$, $\mathbf{s}_{i-1}^{(s)} \sim q(\mathbf{s}_{i-1})$, $\mathbf{s}_i^{(s)} \sim q(\mathbf{s}_i)$ and

$$\mathbf{x}_i^{(s)} := \mathbf{x}^{(s)}(t_i;\mathbf{s}_{i-1}) = \mathbf{s}_{i-1}^{(s)} + \int_{t_{i-1}}^{t_i} \mathbf{f}^{(s)}(\mathbf{x}(\tau))d\tau. \tag{80}$$

S2 EXPERIMENTAL DETAILS

Algorithm 1 GPODEs : Bayesian inference of ODEs using Gaussian processes

Inputs:
- Observed states $\mathbf{Y}$, observation time sequence $\mathbf{t}$.

Initialize hyperparameters:
- Kernel parameters θ, likelihood parameters, inducing locations $\mathbf{Z}$.

Initialize variational parameters:
- Parameters of $q(\mathbf{U}) = \mathcal{N}(\mathbf{m}, \mathbf{Q})$.
- Parameters of $q(\mathbf{x_0}) = \mathcal{N}(\mathbf{a_0}, \mathbf{\Sigma_0})$.

Optimization:

for every optimization step **do**

(1) Sample a function $\mathbf{f}$ from the ODE posterior in (11) by taking following samples:
- Parameters of Fourier bases ω_θ proportional to the spectral density of GP kernel,
- Weights $\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$,
- Sample from the inducing posterior $\mathbf{U} \sim \mathcal{N}(\mathbf{m}, \mathbf{Q})$.

(2) Sample initial state $\mathbf{x}_0 \sim \mathcal{N}(\mathbf{a_0}, \mathbf{\Sigma_0})$.

(3) Compute predicted states $\hat{\mathbf{Y}} = \mathrm{ODEsolve}(\mathbf{f}, \mathbf{x_0}, \mathbf{t})$.

(4) Compute ELBO from (26) : likelihood$(\mathbf{Y}, \hat{\mathbf{Y}})$, $\mathrm{KL}[q(\mathbf{U})||p(\mathbf{U})]$, $\mathrm{KL}[q(\mathbf{x_0})||p(\mathbf{x_0})]$.

(5) Update all parameters with stochastic gradients of ELBO.

end for

S2.1 OPTIMIZATION SETUP

We use Adam (Kingma and Ba, 2014) optimizer and jointly train all the variational parameters and hyperparameters. The complete list of optimized parameters, along with additional method-specific details, are given below.

Vanilla GPODE model. We use 'whitened' representation for the inducing variables and optimize following parameters against the evidence lowerbound (see algorithm 1).

- Variational parameters:

  - Inducing variables $q(\mathbf{U})$, initial states $q(\mathbf{x}_0)$
- Hyperparameters:
  - Inducing locations $\mathbf{Z}$
  - Likelihood parameters: scale parameter for the Gaussian likelihood
  - Kernel parameters: length scales and signal variance parameters in case of squared exponential kernel

Shooting GPODE model. We use 'whitened' representation for inducing variables and optimize the following parameters against the evidence lower bound.

- Variational parameters:
  - Inducing variables $q(\mathbf{U})$, shooting states $q(\mathbf{S})$
- Hyperparameters:
  - Inducing locations $\mathbf{Z}$
  - Likelihood parameters: scale parameter for the Gaussian likelihood
  - Kernel parameters: length scales and signal variance parameters in case of squared exponential kernel

npODE model. We use 'whitened' representation for inducing variables, maximum a posteriori (MAP) objective, and optimize following parameters:

- Inducing values $\mathbf{U}$ and locations $\mathbf{Z}$.
- Likelihood parameters: scale parameter for the Gaussian likelihood.
- Kernel parameters: length scales and signal variance parameters in case of the squared exponential kernel.

NeuralODE model. We use `tanh` activation and a fully connected block with one hidden layer having 32 units in Van der Pol/ Fitz-Hugh Nagumo experiments. In MoCap experiments, we try one/two hidden layers with 64/128 hidden units, and report the best results. All the network parameters were optimized against `MSE` loss.

Bayesian NeuralODE model. We utilized the codebase [2] provided by Dandekar et al. (2020) for training Bayesian version of NeuralODEs. We used networks with one hidden layer and 32 units VDP/FHN experiments and performed posterior sampling with HMC. In case of experiments with long sequences (shooting illustration on VDP and MocCap) the HMC sampling had convergence issues, hence we performed variational inference instead. In case of MoCap experiments, we tried networks with two hidden layers and 64/128 hidden units, and performed mean-field variational inference.

S2.2 ADDITIONAL DETAILS ON THE INDUCING VARIABLES

'Whitening' the inducing variables. While performing sparse inference for GPs using inducing variables, it is a common practice to use noncental parameterization $\tilde{\mathbf{U}} = \mathbf{L}_\theta \mathbf{U}$ where $\mathbf{L}_\theta {\mathbf{L}_\theta}^T = \mathbf{K}_\theta(\mathbf{Z}, \mathbf{Z})$ (Hensman et al., 2015). Such a reparametrization turns the inference for $\mathbf{U}$ with prior $\mathcal{N}(\mathbf{0}, \mathbf{K}_{\mathbf{ZZ}})$ into inference for $\tilde{\mathbf{U}}$ with isotropic Gaussian prior $\mathcal{N}(\mathbf{0}, \mathbf{I})$. This generally improves the optimization performance by decorrelating the latent parameters from each other.

Initializing inducing variables using data gradients. In case of sparse Gaussian process model with inducing variables, we initialize the vector field with empirical gradients from the observed data. We first initialize inducing locations $\mathbf{Z}$ as `kmeans` cluster centers of observations $\mathbf{Y}$. Next we compute empirical gradient estimates, $\dot{\mathbf{Y}} = (\mathbf{y}_2 - \mathbf{y}_1, \mathbf{y}_3 - \mathbf{y}_2, \ldots, \mathbf{y}_N - \mathbf{y}_{N-1})$ at locations $\tilde{\mathbf{Y}} = (\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_{N-1})$ and initialize inducing values $\mathbf{U}$ as the GP mean interpolation of empirical gradients at inducing locations.

$$\mathbf{U} = \Delta t \cdot K(\mathbf{Z}, \tilde{\mathbf{Y}}) K(\tilde{\mathbf{Y}}, \tilde{\mathbf{Y}})^{-1} \dot{\mathbf{Y}}, \tag{81}$$

where Δt is the time difference between two consecutive observations in the dataset.

S2.3 ADDITIONAL DETAILS ON THE CMU MOCAP EXPERIMENT

Details on the dataset. The dataset used in this experiment was obtained from `http://mocap.cs.cmu.edu/`. The database consists of sensor recordings of multiple activities for different subjects in `.amc` files. We selected three subjects

[2] `https://github.com/RajDandekar/MSML21_BayesianNODE`

Table 1: For each subject (a), we report the activity considered for the experiment (b), the data split train/validation/test (c), the number of sequences considered for the corresponding split (d), and the files used in the corresponding split (e).

(a) subject	(b) activity	(c) split	(d) # sequences	(e) files
subject 09	running	train	6	05.amc, 06.amc, 07.amc, 08.amc, 09.amc, 11.amc
		validation	2	01.amc, 02.amc
		test	2	03.amc, 04.amc
subject 35	walking	train	16	01.amc, 02.amc, 03.amc, 04.amc, 05.amc, 06.amc, 07.amc, 08.amc, 09.amc, 10.amc, 11.amc, 12.amc, 13.amc, 14.amc, 15.amc, 16.amc
		validation	3	28.amc, 29.amc, 30.amc
		test	4	31.amc, 32.amc, 33.amc, 34.amc
subject 39	walking	train	6	01.amc, 02.amc, 07.amc, 08.amc, 09.amc, 10.amc
		validation	2	03.amc, 04.amc
		test	2	05.amc, 06.amc

with the most number of walking or running sequences: subjects 09, 35, and 39. The .amc files considered for train, validation and test purposes are given in table 1. The training sequences and their lengths were selected to include at least one full cycle of the dynamics while learning the model. The observation sequence lengths for training/test/validation splits are reported in table 2.

Details on the PCA In the CMU MoCap experiment, we project the data from D dimensional observation-space to $K < D$ dimensional latent-space using eigenvectors corresponding to top-K eigenvalues. The ODE model is then learnt in the latent-space and model predictions are projected back into the observation-space using K eigenvectors. We refer to this as 'inverting the PCA' in the main text.

Table 2: For each subject (a), we report the experiment type (b), the data split train/validation/test (c), and the number of observations considered for the corresponding split.

(a) subject	(b) experiment	(c) split	(d) sequence length
subject 09	short	train	50
		validation	120
		test	120
	long	train	100
		validation	120
		test	120
subject 35	short	train	50
		validation	300
		test	300
	long	train	250
		validation	300
		test	300
subject 39	short	train	100
		validation	300
		test	300
	long	train	250
		validation	300
		test	300

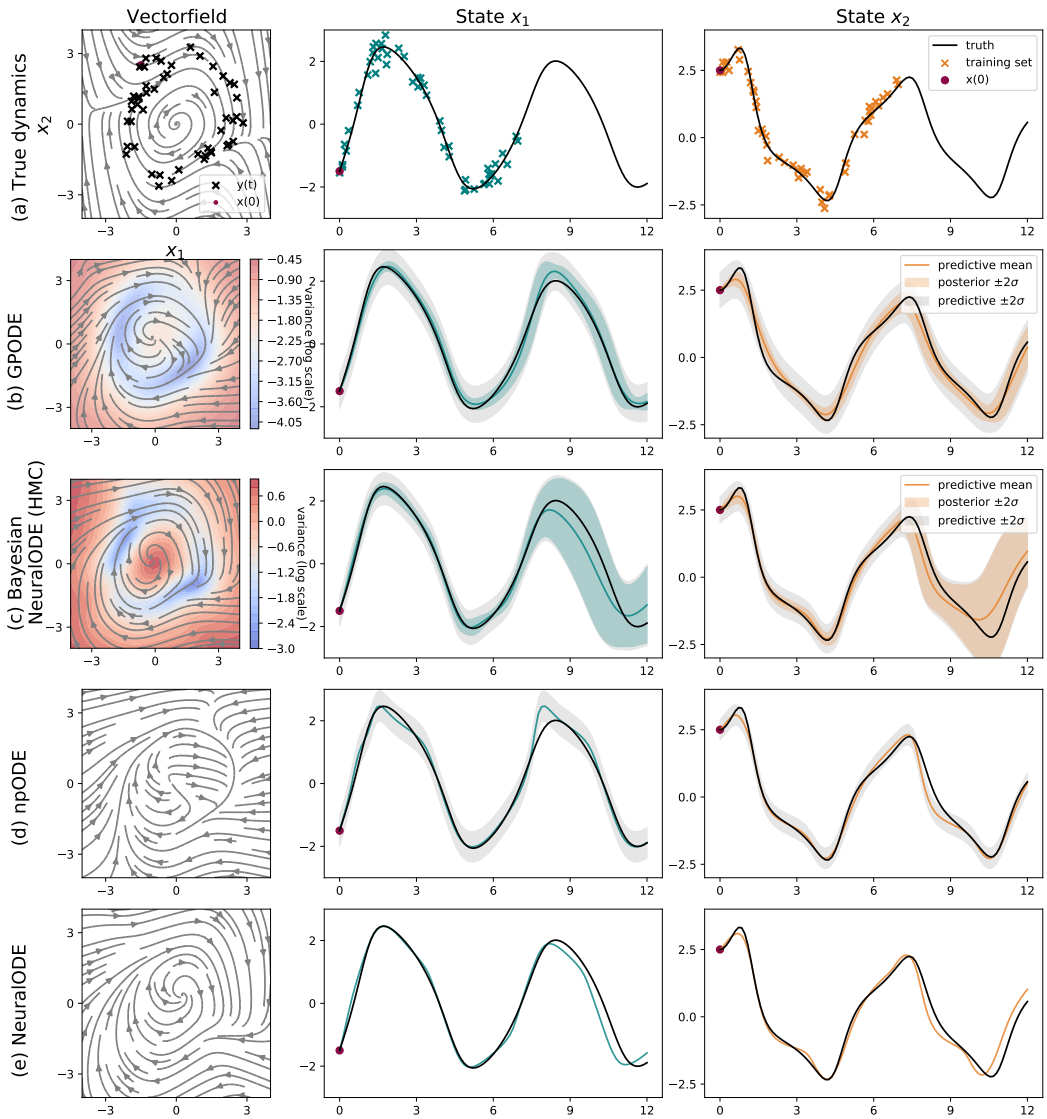


Figure 2: Learning the 2D Van der Pol dynamics on irregularly sampled observations **(a)** with alternative methods **(b-d)**. Column 1 shows the vector fields while columns 2 and 3 show the state trajectories $x_1(t)$ and $x_2(t)$. GPODE learns the posterior accurately.