Exact Inference for Continuous-Time Gaussian Process Dynamics

Katharina Ensinger [1,2 †], **Nicholas Tagliapietra** [1,†,*], **Sebastian Ziesche** [1], **Sebastian Trimpe** [2]

[1] Bosch Center for Artificial Intelligence, Renningen, Germany
[2] Institute for Data Science in Mechanical Engineering, RWTH Aachen University
katharina.ensinger@bosch.com

Abstract

Many physical systems can be described as a continuous-time dynamical system. In practice, the true system is often unknown and has to be learned from measurement data. Since data is typically collected in discrete time, e.g. by sensors, most methods in Gaussian process (GP) dynamics model learning are trained on one-step ahead predictions. While this scheme is mathematically tempting, it can become problematic in several scenarios, e.g. if measurements are provided at irregularly-sampled time steps or physical system properties have to be conserved. Thus, we aim for a GP model of the true continuous-time dynamics. We tackle this task by leveraging higher-order numerical integrators. These integrators provide the necessary tools to discretize dynamical systems with arbitrary accuracy. However, most higher-order integrators require dynamics evaluations at intermediate time steps, making exact GP inference intractable. In previous work, this problem is often addressed by approximate inference techniques. However, exact GP inference is preferable in many scenarios, e.g. due to its mathematical guarantees. In order to enable direct inference, we propose to leverage multistep and Taylor integrators. We demonstrate how exact inference schemes can be derived for these types of integrators. Further, we derive tailored sampling schemes that allow one to draw consistent dynamics functions from the posterior. The learned model can thus be integrated with arbitrary integrators, just like a standard dynamical system. We show empirically and theoretically that our approach yields an accurate representation of the continuous-time system.

1 Introduction

Many systems can be described by a continuous-time ordinary differential equation (ODE)

$$\dot{x}(t) = f(x(t)) \text{ with } f : \mathbb{R}^d \to \mathbb{R}^d. \tag{1}$$

Dynamics model learning deals with the problem of estimating the dynamics function f from data. Usually, it is not possible to measure the states $(x_n)_{n=1\dots N}$ in continuous time, but noisy measurements $(\hat{x}_n)_{n=1\dots N}$ at (potentially irregularly-sampled) discrete time points t_n can be obtained by sensors, where

$$\hat{x}_{n,u} = x_{n,u} + \nu_{n,u}, \tag{2}$$

$\nu_{n,u} \sim \mathcal{N}(0, \sigma_u^2)$, $n = 1 \dots N$ and $u = 1 \dots d$. Gaussian processes (GPs) are a powerful probabilistic framework and can be interpreted as a distribution over functions. Hence, training a GP is a popular approach in dynamics model learning, providing uncertainty estimates for the model. Typically, a GP model $\tilde{f}$ is trained on one-step ahead predictions (Deisenroth and Rasmussen 2011; Doerr et al. 2018)

$$x_{n+1} = \tilde{f}(x_n). \tag{3}$$

Mathematically, this structure is tempting since it can be addressed via standard GP regression, where input-output pairs are given as neighboring points in the trajectory. Thus, the GP can be directly conditioned on the data. However, the approach can lead problems in various scenarios. Especially, it is not suitable for irregularly-sampled or missing training data or if predictions at intermediate time steps are required. Further, physical structure of the true system, such as energy or volume is typically not preserved (Ensinger et al. 2022).

We address this problem by learning GP dynamics that represent f more accurately. To this end, we leverage numerical integrators, in particular higher-order multistep and Taylor integrators. Numerical integrators provide the necessary tools to approximate the solution of system (1) with arbitrary accuracy for known dynamics f. However, we can leverage the approximation qualities of numerical integrators in a learning-based scenario, where the true dynamics are unknown. In particular, applying a higher-order integrator to the unknown system Eq. (1) yields a more accurate approximation of the continuous-time dynamics f. From a numerical perspective, one-step ahead predictions (3) correspond to the explicit Euler method. This follows by identifying $\tilde{f}(x_n)$ with $x_n + (x_{n+1} - x_n)f(x)$. Being a method of order one, this indicates that the learned dynamics typically do not provide an accurate representation of the continuous-time system.

Generalizing to higher-order integrators raises technical difficulties. In particular, most integrators require dynamics evaluations at intermediate time steps (Hairer, Nørsett, and Wanner 1987) making standard GP inference intractable. In previous works, this problem is often addressed by approximating the GP posterior with variational inference (Hegde

[†] Authors contributed equally to this work

[*] Work was conducted at Bosch Center for Artificial Intelligence during master's studies at University of Padova.

et al. 2022; Ensinger et al. 2022). However, variational inference has some downsides. Especially mathematical guarantees of the GP, such as error bounds, get lost. The beneficial structure of (varying step-size) multistep and Taylor integrators enables us to derive exact inference schemes. This is due to the fact that the dynamics are evaluated only at past and current points in the trajectory for these integrators. We are thus able to maintain the mathematical properties of standard GP dynamics model learning (cf. Eq. (3)) while learning an accurate ODE model.

On a technical level, we derive a flexible framework that automatically computes tailored GP kernels from a given time discretization and integration scheme. These kernels are then used for inference. Further, we derive corresponding decoupled sampling (DS) schemes based on (Wilson et al. 2020). This technique enables sampling full dynamics functions from the GP posterior. In contrast to standard GP sampling, is is not required to condition dynamics evaluations on previous ones. Thus, a trajectory sample is simply obtained by sampling a dynamics function from the posterior and integrating it numerically. We show theoretically and empirically that we are able to learn good ODE representations by training with integrators of sufficient order. In summary, our contributions are

- a method that allows for learning GP-based ODE dynamics via standard GP inference enabling the computation of error bounds between true and learned dynamics;
- a framework that allows for computing the corresponding kernels for multistep and Taylor integrators of arbitrary order and potentially irregularly-sampled time points; and
- a decoupled sampling scheme that allows one to sample consistent dynamics from the GP posterior.

2 Technical Background

We provide a summary of the necessary mathematical tools.

2.1 Gaussian Process Regression

A GP is a random function $g : \mathbb{R}^d \to \mathbb{R}$ (Rasmussen and Williams 2005). Similar to a normal distribution, a GP is determined by its mean function $m(x) := \mathbb{E}[g(x)]$ and covariance function $k(x, y) := \text{cov}(g(x), g(y))$. Here, we assume $m(x) = 0$. Conditioning a GP on observations $Y = (Y_n)_{n=1...N}$ where $Y_n \in \mathbb{R}$, inputs $X = (X_n)_{n=1...N}$ where $X_n \in \mathbb{R}^d$ and assuming observation noise distributed with $\mathcal{N}(0, \lambda)$, yields a normal distributed predictive posterior distribution. For M test points $(x^\star_m)_{m=1...M}$, we obtain $K := (k(X_n, X_m))_{n,m=1...N}$, $k(x^\star, x^\star) := (k(x^\star_n, x^\star_m))_{n,m=1...M}$ and $k(x^\star) := (k(x^\star_m, X_n))_{m=1...M,n=1...N}$. Then, it holds that $f(x^\star) \sim \mathcal{N}(\mu(x^\star), \Sigma(x^\star))$ with

$$\begin{aligned} \mu(x^*) &= k(x^\star)^T (K + \lambda I)^{-1} Y \\ \Sigma(x^\star) &= k(x^\star, x^\star) - k(x^\star)^T (K + \lambda I)^{-1} k(x^\star). \end{aligned} \tag{4}$$

Training: During training, we parametrize k with θ and write $k_\theta(x, y)$. The trainable parameters θ and λ are adapted by maximizing the log probability of the observations

$$p(Y) = \mathcal{N}(Y|0, K_\theta + \lambda I). \tag{5}$$

Remark: In this work, we learn a dynamics function with d-dimensional outputs. This problem is addressed by learning d separate GPs with individual trainable parameters.

Decoupled sampling (DS): Standard GP conditioning allows one to evaluate the posterior at a finite subset of points. However, trajectory sampling from a learned vector field is an iterative process, where subsequent trajectory points depend on previous ones. This requires conditioning on previous samples (Hewing et al. 2020), which is intractable for long trajectories and complex integrators. To address this problem, decoupled sampling (DS) offers a technique for efficiently drawing consistent functions from the GP posterior (Wilson et al. 2020). On a technical level, it decomposes the posterior via Matheron's rule (Howarth 1979). A sample from the posterior is then obtained by combining a sample from the prior with a deterministic update. The prior can be approximated via a finite-dimensional representation with random basis functions and weights. In summary it holds that

$$\begin{aligned} g(x^\star|Y) \approx &\sum_{i=1}^{S} w_i \phi_i(x^\star) + k(x^\star)(K + \lambda I)^{-1} \\ &(Y - \sum_{i=1}^{S} w_i \Phi_i - \epsilon) \end{aligned} \tag{6}$$

for the posterior conditioned on observations. Here, the stationary GP prior is represented via S Fourier bases ϕ_i, $w_i \sim \mathcal{N}(0, 1)$ and $\Phi_i = (\Phi_{i,n})_{n=1...N} \in \mathbb{R}^N$ with $\Phi_{i,n} = \phi_i(X_n)$ (Rahimi and Recht 2008). Further, $\epsilon \sim \mathcal{N}(0, \lambda I)$.

2.2 Numerical Integration

Numerical integrators for an ODE (1) compute an approximation $\bar{x}_n$ of the solution $x(t_n)$ at discrete time steps t_n (Hairer, Nørsett, and Wanner 1987). Among other properties, they are determined by their order (of consistency) P. Mathematically, the order corresponds to the truncation index of the Taylor series up to which the correct solution and the approximate solution coincide. Therefore, a higher order typically leads to a more accurate approximation $\bar{x}_n$. Here, we use the notation $\bar{x}_n$ to indicate the subtle difference to ground truth states x_n.

Varying step-size multistep integrators: Multistep integrators approximate the solution $x(t_{n+M})$ by taking the last $M - 1$ points $\bar{x}_n, \dots \bar{x}_{n+M-1}$ into account. It holds that

$$\sum_{j=0}^{M} a_{jn} \bar{x}_{n+j} = \sum_{j=0}^{M} b_{jn} f(\bar{x}_{n+j}), \tag{7}$$

where the coefficients $A = (a_{jn})^{n=0,\dots,N}_{j=0,\dots,M}$ and $B = (b_{jn})^{n=0,\dots,N}_{j=0,\dots,M}$ depend on the step sizes $h_n = t_{n+1} - t_n$ and $(\bar{x}_0, \dots, \bar{x}_M) = (x(t_0), \dots x(t_M))$. In case $b_{Mn} \neq 0$, the system is implicit, and thus a minimization problem has to be solved for $\bar{x}_{n+M}$. The parameters $a_{jn}, b_{jn} \in \mathbb{R}$ determine the properties of the method, e.g., the order P. For constant step sizes, the coefficients a_{jn} and b_{jn} reduce to a_j and b_j.

Taylor integrators: Taylor integrators are based on a Taylor expansion of the solution $x(t+h)$ around $x(t)$. Truncating

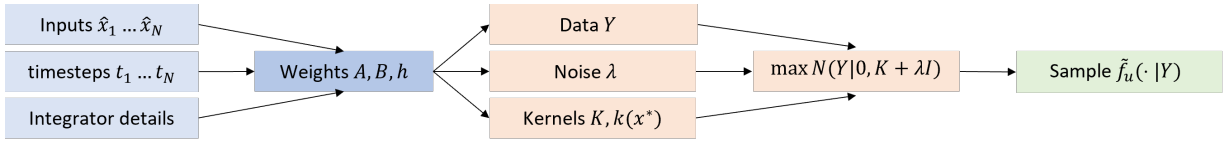

Figure 1: Overview over training and predicting for dimension u. Data $\hat{x}_n$, time steps t_n and integrator details are used to generate the integrator coefficients (blue). This allows one to compute all necessary components for training, including transformed observations Y, corresponding noise λ and kernels (orange). After training, we obtain $\tilde{f}_u$ via DS (green).

the expansion at index P yields the Taylor integrator of order P. This results in

$$\bar{x}_{n+1} = \bar{x}_n + \sum_{l=1}^{P} \frac{h_n^l}{l!} f^l(\bar{x}_n), \tag{8}$$

where $f^l : \mathbb{R}^d \to \mathbb{R}^d$, $f^1 = f, f^2 = f'f, \dots, f^{k+1} = \frac{d}{dx} f^k f$ and $\bar{x}_0 = x(t_0)$ (Abad et al. 2012). The Taylor integrator of order 1 corresponds to the explicit Euler method. In practice, f^l is often too complex to compute since it requires the computation of higher-order derivatives of the dynamics f. However, in a learning-based scenario, the approach is beneficial as we shall see.

3 Method

Next, we develop the technical details of merging GPs with multistep and Taylor integrators.

Main idea and setup: Our goal is to learn an accurate representation of the true *continuous-time* dynamics f from discrete-time trajectory data $\hat{x}_n$ via exact inference. To this end, we apply appropriate numerical integrators, such as multistep and Taylor integrators to the unknown dynamical system (cf. Eq. (1)) and train a GP model for the dynamics. In the multistep case, we aim for GP dynamics $\tilde{f}$ that fulfill $\sum_{j=0}^{M} a_{jn} x_{n+j} = \sum_{j=0}^{M} b_{jn} \tilde{f}(x_{n+j})$. In the Taylor case, we aim for GP dynamics $\tilde{f}^l$ that fulfill $x_{n+1} = x_n + \sum_{l=1}^{P} \frac{h^l}{l!} \tilde{f}^l(x_n)$. Intuitively, the higher the order, the more accurate the approximation.

Common setup: We model each dimension of the dynamics via separate GPs $\tilde{f}_u$ with kernel function k in the multistep case for $u = 1 \dots d$ and $\tilde{f}_u^l$ with kernel function k_l in the Taylor case for $u = 1 \dots d$ and $l = 1 \cdots = P$. For brevity, we omit the index u for the kernels. The full dynamics are obtained via $\tilde{f} = (\tilde{f}_1 \dots \tilde{f}_d)^T$, respectively $\tilde{f}^l = (\tilde{f}_1^l \dots \tilde{f}_d^l)^T$. An overview of the method is given in Fig. 1, which we will detail in the following. We do not model input noise as it is typical for GP dynamics model learning (Deisenroth and Rasmussen 2011). Therefore, we treat the noisy data $\hat{x}_n$ as inputs. However, input noise could be incorporated as proposed by Mchutchon and Rasmussen (2011).

3.1 Data Processing and Kernels

In contrast to the standard scenario, the training data are not given as input-output pairs. Instead, linear combinations of dynamics evaluations correspond to linear combinations of trajectory points (cf. Eq. (7) and (8)). Therefore, the GP observations Y are a linear combination of trajectory points. Here, we approximate the corresponding time-varying noise $\epsilon \sim \mathcal{N}(0, \lambda I)$ with a diagonal covariance matrix for computational efficiency and stability. However, our framework provides the option to consider correlated noise between different observations. See the appendix for details. Similar to the standard regression setting, we compute the covariance matrices K between training inputs and $k(x^\star)$ between training and test inputs. Since all calculations are done dimension wise, we obtain d different representations of K and $k(x^\star)$.

Multistep integrators: We obtain observations $Y = (Y_n)_{n=1\dots N-M}$, where $Y_n = \sum_{j=0\dots M} a_{jn} \hat{x}_{n+j,u}$ by applying the left-hand side of Eq. (7) to $\hat{x}_{n+j,u}$. Thus, it holds that $Y_n = \sum_{j=0}^{M} b_{jm} \tilde{f}_u(\hat{x}_{n+j}) + \epsilon_n$ with noise $\epsilon = (\epsilon_n)_{n=1\dots N-M}$, corresponding $\lambda = (\lambda_n)_{n=1\dots N-M}$ and $\lambda_n = \sum_{j=0}^{M} a_{jn}^2 \sigma_u^2$. For $x^\star \in \mathbb{R}^d$, we obtain $K \in \mathbb{R}^{N-M \times N-M}$ and $k(x^\star) \in \mathbb{R}^{N-M}$, where

$$\begin{aligned}(K)_{nm} &= \operatorname{cov}\left(\sum_{j=0}^{M} b_{jn} \tilde{f}_u(\hat{x}_{n+j}), \sum_{j=0}^{M} b_{jm} \tilde{f}_u(\hat{x}_{n+j})\right) \\ &= \sum_{i=0}^{M} \sum_{j=0}^{M} b_{jn} k(\hat{x}_{n+j}, \hat{x}_{m+j}) b_{im}\end{aligned} \tag{9}$$

and

$$\begin{aligned}(k(x^\star))_n &= \operatorname{cov}\left(\tilde{f}_u(x^\star), \sum_{j=0}^{M} b_{jn} \tilde{f}_u(\hat{x}_{n+j})\right) \\ &= \sum_{j=0}^{M} b_{jn} k(x^\star, \hat{x}_{n+j}).\end{aligned} \tag{10}$$

Taylor integrators: We obtain observations $Y = (Y_n)_{n=1\dots N-1}$, where $Y_n = \hat{x}_{n+1,u} - \hat{x}_{n,u}$ Thus it holds that $Y_n = \sum_{l=1}^{P} \frac{h_n^l}{l!} \tilde{f}_u^l(\hat{x}_n) + \epsilon_n$ with $\epsilon = (\epsilon_n)_{n=1\dots N-1}$ and corresponding $\lambda = 2\sigma_u^2$ (cf. Eq. (8)). In order to make inference tractable, we approximate each dimension u of the truncated Taylor representation of order P (cf. Eq. (8)) with P independent GPs $\tilde{f}_u^l : \mathbb{R}^d \to \mathbb{R}$ and kernels k_l, where $l = 1, \dots, P$. Despite this approximation, we are able to derive error bounds. With the independence assumption, we obtain for $x^\star \in \mathbb{R}^d$ that $K \in \mathbb{R}^{N-1 \times N-1}$ and $k_i(x^\star) \in \mathbb{R}^{N-1}$,

where

$$(K)_{nm} = \operatorname{cov}\left(\sum_{l=1}^{P} \frac{h_n^l}{l!}\tilde{f}_u^l(\hat{x}_n), \sum_{l=1}^{P} \frac{h_m^l}{l!}\tilde{f}_u^l(\hat{x}_m)\right) = \sum_{l=1}^{P} \frac{h_n^l h_m^l}{l!l!} k_l(\hat{x}_n, \hat{x}_m) \tag{11}$$

and

$$(k_i(x^\star))_n = \operatorname{cov}\left(\tilde{f}_u^i(x^\star), \sum_{l=1}^{P} \frac{h_n^l}{l!}\tilde{f}_u^l(\hat{x}_n)\right) = \frac{h_n^i}{i!} k_i(x^\star, \hat{x}_n). \tag{12}$$

In our experiments, we train separate kernels and hyperparameters k_l, which is computationally efficient due to parallelization. However, our framework also offers the option to compute adapted kernels. Intuitively, the kernels and hyperparameters representing all higher-order derivatives depend on each other (see Appendix Sec. 1.2 for details).

Training and inference: Analogous to the standard scenario, we parametrize the kernel functions k and k_l with θ. The trainable parameters θ and the noise parameter σ_u are adapted by maximizing Eq. (5). Here, K is obtained via Eq. (9) in the multistep and via Eq. (11) in the Taylor case. Similarly, calculating the posterior distribution is now straightforward. In the multistep case, we obtain the posterior distribution for $f_u(x^\star)$ by applying Eq. (4) to the corresponding transformed Y, λ and the kernels in Eq. (9). In the Taylor case, we obtain the posterior distribution for $f_u^i(x^\star)$ by applying Eq. (4) to the corresponding transformed Y, λ and kernels in (11). Here, $k(x^\star)$ in Eq. (4) is replaced by $k_i(x^\star)$.

3.2 Sampling

Standard GP sampling via Eq. (4) is computationally not tractable for sampling trajectories, being an iterative process. This is due to the fact that all dynamics evaluations have to be conditioned on previous ones. To make trajectory sampling feasible, we derive a DS scheme similar to Eq. (6) for multistep and Taylor integrators by applying Matheron's rule (Howarth 1979) to the posterior.

Multistep: We sample from $\tilde{f}_u$ via

$$\tilde{f}_u(\cdot|Y) \sim \sum_{k=1}^{S} w_k \phi_k(\cdot) + k(\cdot)(K+\lambda I)^{-1} \left(Y - \sum_{k=1}^{S} w_k F_k - \epsilon\right), \tag{13}$$

with $F_k = (F_{k,n})_{n=1\dots N} \in \mathbb{R}^{N-M}$ and $(\tilde{F}_{k,n}) = \sum_{j=0}^{M} b_{jn}\phi_k(X_n)$. Here, samples from the prior $\tilde{f}_u$ are represented by S random Fourier bases ϕ_k and corresponding weights w_k (cf. Eq. (6)). The time-varying noise is represented by $\epsilon \sim \mathcal{N}(0, \lambda I)$.

Taylor: We sample from the Taylor component $\tilde{f}_u^i$ via

$$\tilde{f}_u^i(\cdot|Y) \sim \sum_{k=1}^{S} w_{ki}\phi_{ki}(\cdot) + k_i(\cdot)(K+\lambda I)^{-1} \left(Y - \sum_{k=1}^{S}\sum_{l=1}^{P} w_{kl}\tilde{F}_{kl} - \epsilon\right), \tag{14}$$

with $\tilde{F}_{kl} = (\tilde{F}_{kl,n})_{n=1\dots N} \in \mathbb{R}^{N-M}$ and $\tilde{F}_{kl,n} = \frac{h_n^l}{l!}\phi_{kl}(X_n)$. We represent the prior $\tilde{f}_u^l, l = 1, \dots, P$ via S random Fourier bases ϕ_{kl} and weights w_{kl} (cf. Eq. (6)). The noise is represented by $\epsilon \sim \mathcal{N}(0, \lambda I)$. We obtain the full truncated Taylor series for dimension u by sampling from $\tilde{f}_u^l(\cdot|Y), l = 1, \dots, P$. See Appendix Sec. 1.1.2 for detailed derivations of kernels and prediction schemes.

3.3 Predictions

The proposed DS schemes allow treating the trained model like a standard ODE since full dynamics functions can be sampled from the posterior and evaluated at arbitrary points. In particular, we are able to integrate the learned model with arbitrary integrators including adaptive step size ones (Hairer, Nørsett, and Wanner 1987). These methods integrate system (1) with arbitrary accuracy leading to a negligible numerical error. We can therefore leverage them to evaluate if our learned dynamics $\tilde{f}$ is indeed a good ODE model similar to (Ott et al. 2021). To this end, the error is evaluated with an adaptive step-size integrator on a train or validation set indicating whether the model represents the ODE accurately. If not, it is retrained with higher accuracy. In the experiments, we demonstrate that multistep integrators are suitable to learn continuous-time dynamics from regularly and irregularly-sampled grids. Taylor integrators, however, are especially suitable for varying step sizes. This is due to the fact that the Taylor representation (8) does not force the GPs $\tilde{f}_u^l$ to learn the correct part of the Taylor series since the step size has no influence on the learning task in the regularly-sampled case. However, due to the uniqueness of the Taylor series, varying step sizes limit the freedom of the GPs. Even if this is not the focus of this paper, our method is also useful to obtain structure-preserving predictions as addressed in (Ensinger et al. 2022). Embedding the physical structure (e.g. Hamiltonian) in the GP kernel and predicting the trained model with a structure-preserving integrator (e.g. a symplectic one) yields predictions that are for example volume-preserving.

3.4 Error Estimates for Multistep Methods

We aim to quantify the accuracy of our learned dynamics by bounding $\|f_u(x) - \mu(x)\|$, where $u = 1 \dots d$ and $x \in \mathbb{R}^d$ and $\mu(x)$ the mean approximation. Here, $f = (f_1, \dots, f_d)^T$ denotes the true ODE dynamics (cf. Eq. 1), $\mu(x) = k(x)^T(K+\lambda I)^{-1}Y$ the posterior mean and $\sigma^2(x) = k(x,x) - k(x)^T(K+\lambda I)^{-1}k(x)$ the posterior variance, where Y, K and $k(x^\star)$ are calculated as described in Sec. 3.1. We consider noiseless data and a constant jitter $\lambda > 0$. We assume $f_u \in H_u$, where H_u denotes a reproducing kernel Hilbert space (RKHS) represented by the kernel k. The main

idea is to combine GP error bounds such as Chowdhury and Gopalan (2017) or Fiedler, Scherer, and Trimpe (2021) with the error of the integrators.

Theorem 1 (Multistep error) *Consider a multistep method of order P with coefficient matrices A and B (cf. Eq. (7)). Assume $f_u \in H_u$ with kernel k and RKHS norm $\|f_u\|_k \leq C$. Further, assume that $|f_u^{P+1}|_\Omega \leq L$ and $|f_u^{P+2}I|_\Omega \leq L$. Under mild assumptions and with $\lambda = 1 + \tau$ it holds*

$$\|f_u(x) - \mu(x)\| \leq \sigma(x)\sqrt{\|((K + \tau I)^{-1} + I)^{-1}\|} \\ (C + \text{Constant}(N, M, \max(h), A, B, P, L, \lambda)). \tag{15}$$

Thm. 1 enables estimating the model error based on its uncertainty estimates. Intuitively, a small GP uncertainty $\sigma(x)$ together with a small RKHS norm of the true dynamics and an accurate integrator yields accurate ODE dynamics. We provide details, proofs, and a similar result for Taylor integrators in Appendix Sec. 2.

4 Related Work

Dynamics model learning is a broad field and has been addressed by different communities for many years (Nguyen-Tuong and Peters 2011; Ljung 1999). Most GP dynamics model learning approaches consider discrete-time systems (Doerr et al. 2018; Deisenroth and Rasmussen 2011). While discrete-time systems are typically modeled with one-step ahead predictions or history-based approaches, there are also approaches that apply numerical integrators (Ensinger et al. 2022; Rath et al. 2021; Brüdigam et al. 2022). However, in contrast to this work, they focus on the preservation of physical structure.

Continuous-time models are often learned with neural ODEs (Chen et al. 2018). To this end, the dynamics are modeled with a neural network and integrated with an integrator of desired accuracy. Zhu et al. (2022) provide error estimates for the accuracy of neural ODEs trained on Runge-Kutta integrators. Neural networks have been trained on multistep integrators as well (Raissi, Perdikaris, and Karniadakis 2018) followed by error analysis (Keller and Du 2021). The most recent work bounds the error based on inverse modified differential equations (Zhu, Wu, and Tang 2022). However, none of these approaches leverages varying step size multistep methods. Thus, the approaches and results can not be applied to irregularly-sampled data. Djeumou et al. (2022) leverage Taylor integrators for neural ODE predictions. However, in contrast to this work, they do not leverage them for training. Further, all of the above approaches refer to deterministic neural networks and thus do not provide uncertainty estimates. Instead, we aim for a probabilistic GP model that can be trained and updated via exact inference and provides error bounds. Multistep methods have been combined with GPs in Teymur, Zygalakis, and Calderhead (2016); Raissi, Perdikaris, and Karniadakis (2017). However, they address the probabilistic numerics setting, where the dynamics are known. Instead, we consider the problem of inferring unknown dynamics from data.

Some works combine ODE learning with GPs. Gradient matching methods model the measured trajectories with a GP (Wenk et al. 2019; Dondelinger, Rogers, and Husmeier 2013). However, they consider a parametric dynamics model with unknown parameters for the dynamics. The concept is extended in Heinonen and d'Alché Buc (2014) to nonparametric models. However, they still consider the gradient matching approximation and the learned dynamics are not a GP. Some approaches also model ODE dynamics with GPs. Heinonen et al. (2018); Hegde et al. (2022) tackle the problem by applying sparse GPs and train them similarly to neural ODEs. This requires an approximation via variational inference, while we aim for exact inference. Ridderbusch, Ober-Blöbaum, and Goulart (2023) consider standard GP conditioning as well by linearizing parts of the dynamics systems, which allows to propagate the uncertainty of the approximated system. However, this requires approximations at many points and the uncertainty is not propagated exactly.

5 Experiments

Next, we evaluate our methods numerically and support the intuitive and theoretical findings. Our framework provides the option to train with various integrators of arbitrary order and time irregularity. In contrast to most GP dynamics models, we treat the learned dynamics indeed as an ODE and perform predictions with an integrator that solves the ODE almost exactly. In particular, the prediction integrator does not necessarily correspond to the training integrator. We consider both, mean and DS predictions and conduct experiments with fixed and varying step sizes. We show that (i) for multistep integrators, the higher the order, the better the ODE approximation on regular and irregular grids. (ii) Taylor integrators are especially effective on irregular grids (cf. Sec 3.1). (iii) We can compete with a variational inference baseline. (iv) Through extensive experiments, we investigate which integrator to use in which scenario including their limitations. We further demonstrate that our framework can cope with different choices of integrators.

Integrators: We consider the three main classes of multistep integrators: Adam-Bashforth (AB) methods, representing explicit integrators; and Adam Moulton (AM) and backward-difference formulas (BDF), both representing implicit integrators that require the solution of a minimization problem (see Hairer, Nørsett, and Wanner (1987)). For all integrators, we consider orders (of consistency) 1 to 3. We refer to them as "integrator"+"order". So, Adam-Bashforth of order 1 would be denoted "AB 1". Since Taylor 1 and AB 1 correspond to the explicit Euler, we conduct these experiments only once. Similarly, AM 1 and BDF 1 correspond to the implicit Euler method. For predictions, we consider the Runge-Kutta 4(5) integrator, an adaptive step size integrator that makes the numerical error negligible and solves the ODE (1) almost exactly (Hairer, Nørsett, and Wanner 1987). We refer to it as RK4(5). This allows to quantify whether our model provides an accurate ODE approximation. In Appendix Sec. 3.2, we provide results for training and predictions with identical integrator. On regular grids, this provides comparable results for all integrators. Thus, the differences in accuracy are caused by the ODE approximation qualities.

Experimental setup: For all experiments, we consider DS predictions by drawing independent trajectories from the GP posterior and computing the statistical mean and variance. We evaluate the mean squared error (MSE) between data and predictions on five independent runs and report mean and standard deviation. We consider the explicit Euler (AB 1) as a baseline. We further compare to the GP-ODE proposed in Hegde et al. (2022), a variational inference-based approach that works similarly as a neural ODE. By considering orders 1 to 3 for each integrator, we investigate how the order affects the results. All GPs are modeled with ARD kernels (Rasmussen and Williams 2005). We consider simulated systems and real-world data. In Appendix Sec. 3 and 4, we report details, runtimes and additional results. This includes results for the Taylor integrators with adapted kernels (cf. Sec. 3.1).

5.1 Systems

Next, we specify the systems and learning tasks. Details for the simulated systems are provided in Appendix Sec. 4.1.

Damped harmonic oscillator (DHO): The DHO system represents an oscillator subject to a damping or friction force (Zhu, Wu, and Tang 2022). We consider a timeline with regular step size and generate a trajectory of 10 seconds and step size $h = 0.01$. The first 500 steps are used for training, while predictions are performed on the full trajectory. We consider the multistep integrators AB, AM and BDF, order 1 to 3. We investigate if indeed the accuracy of the ODE increases with the order. The results are displayed in Table 1, an example for DS predictions is Fig. 3 (left).

Van-der-Pol oscillator (VDP): We simulate the VDP system (Cveticanin 2013) on an irregular timeline by sampling the step size within a certain range b via $t_{i+1} = t_i + h(1 + (w - 1/2)b)$, with $w \in \mathcal{U}(0, 1)$ and step size $h = 0.1$. Here, we choose $b = 0.5$. In Appendix Sec. 3.3, we add results for $b = 0.3$. We compute rollouts with 100 steps. Training is performed on the first 50 steps, predictions are performed on the full trajectory. We consider AB, AM and BDF as well as Taylor, order 1 to 3. The results are displayed in Table 2, DS rollouts and phase plots in Fig. 2.

Real spring system: We consider measurements from a linear mass-spring system on an air track from Schmidt and Lipson (2009). The dataset corresponds to a 871 time steps long trajectory where each point is a vector $\mathrm{x} = (x, v)$ of position and velocity. We use the first 400 steps for training, while predictions are performed on the full trajectory. We consider AB, AM and BDF. The results are displayed in Table 3, an example for DS rollouts including GP uncertainty in Fig. 3 (middle).

Human motion data (MoCap): Like Hegde et al. (2022), we consider experimental human motion data from CMU MoCap database for subject 09 short. We also use the same trajectories for training and testing as them. The 50-dimensional data are projected into a 3-dimensional space by applying a PCA. We learn the ODE in the PCA space and project it back to the original space to obtain predictions. This results in some loss of information. However, since we aim to apply exact inference, we can not embed our method in a latent space in contrast to Hegde et al. (2022). The results are displayed in Table 4, a mean rollout in Fig. 3 (right).

5.2 Results

The results demonstrate that accurate ODE models from regular and irregular timelines can be obtained with higher-order integrators while maintaining the advantages of exact GP inference. This also means that the standard learning procedure via explicit Euler (AB 1) does not provide good ODE representations. Further, the results coincide with the theoretical findings and the intuition, i.e. the accuracy increases with the order. However, the approach has limits if the step sizes are too large or too irregular. In practice, the necessary order to obtain an accurate ODE has to be evaluated numerically, e.g. by computing the error with RK4(5) on a validation set (cf. Sec. 3). **DHO:** The results for the DHO system demonstrate that higher-order multistep methods allow learning an ODE that outperforms the GP-ODE in terms of accuracy. Further, the accuracy increases with the order (cf. Fig 2 (left)). While the implicit integrators (AM and BDF) provide accurate results already for order 2, the explicit method (AB) struggles up to order 3. After entering the extrapolation area, the higher-order integrators deviate from the trajectory as well after some time (cf. Fig 2 (left)). This behavior is even stronger for the GP-ODE baseline. However, AB 1 already deviates in the training area. **VDP:** The results for VDP extend these findings to the irregularly-sampled case (cf. Fig. 3). In contrast to the DHO system, the implicit BDF method still causes problems for order 2. Further, Taylor methods work well for varying step size as described in Sec. 3.1. In Appendix 3.2, we demonstrate that here, higher order yields higher accuracy also if we predict with the training integrator. This confirmes that the explicit Euler is not suitable for varying step sizes. **Real-world data:** The findings for the real spring system methods show that we can also learn accurate ODEs from real-world data (cf. Fig 2 (middle)). On the MoCap data, our integrators are outperformed by the GP-ODE when sampling with DS. Also, increasing accuracy with the order is not clearly visible for DS predictions. This might be caused by the high observation noise our method learns. Further, due to the variational inference setting, the GP-ODE can be trained on the latent space balancing information loss caused by the PCA. Thus, the two methods are not fully comparable. In contrast, our mean predictions are accurate and the typical trend is visible (cf. Fig 2 (right)).

Integrator	order	MSE
Baseline GP-ODE	-	0.187 (0.091)
Baseline AB	1	0.750 (0.232)
AB	2	2.62 (0.011)
AB	3	0.062 (0.027)
AM/ BDF	1	1.175 (0.060)
AM	2	0.027 (0.015)
AM	3	0.043 (0.023)
BDF	2	0.009 (0.006)
BDF	3	**0.006** (0.001)

Table 1: DS predictions with RK4(5) for DHO

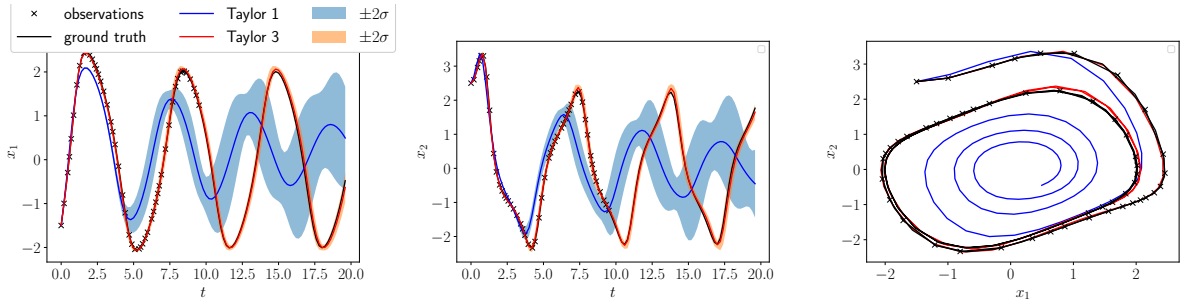

Figure 2: DS predictions (left, middle) and corresponding phase (right) for the VDP system with Taylor order 1 and 3. Shaded regions indicate the GP uncertainty. With increasing order, a clear improvement is visible.

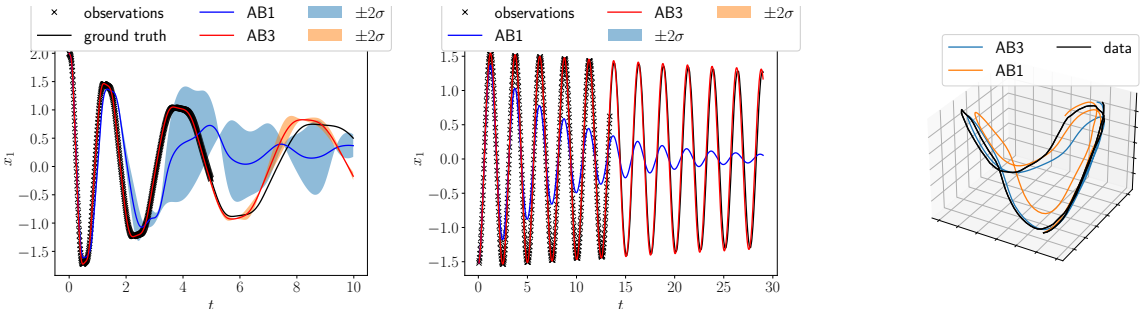

Figure 3: DS predictions on a single seed for the DHO system (left) and the real spring system (middle). Uncertainty is illustrated via shaded regions. Mean predictions in PCA space for MoCap (right). Raising the integrator order yields an improvement.

Integrator	order	MSE
Baseline GP-ODE	-	0.043 (0.028)
Baseline AB	1	1.843 (0.141)
AB	2	0.030 (0.007)
AB	3	0.114 (0.058)
AM/BDF	1	34.578 (18.372)
AM	2	0.015 (0.003)
AM	3	0.016 (0.005)
BDF	2	17.753 (30.677)
BDF	3	0.063 (0.037)
Taylor	2	0.01 (0.004)
Taylor	3	**0.005** (0.002)

Table 2: DS predictions with RK4(5) for VDP

Integrator	order	MSE
Baseline GP-ODE	-	0.163 (0.093)
Baseline AB	1	0.502 (0.029)
AB	2	0.026 (0.037)
AB	3	**0.007** (0.009)
AM/ BDF	1	420.5 (422)
AM	2	0.014 (0.004)
AM	3	0.016 (0.014)
BDF	2	0.050 (0.037)
BDF	3	0.035 (0.041)

Table 3: DS predictions with RK4(5) for real spring system

Integrator	order	type	MSE
Baseline GP-ODE	-	DS	**20.71** (1.25)
Baseline AB	1	DS	40.97 (15.47)
AB	2	DS	55.79 (46.44)
AB	3	DS	30.42 (9.20)
Baseline AB	1	mean	37.76 (1.39)
AB	2	mean	**10.12** (0.33)
AB	3	mean	11.78 (1.58)

Table 4: Mean and DS predictions with RK4(5) for MoCap

6 Conclusion and Future Work

We propose a flexible framework to learn ODEs via exact GP inference from discrete data for both, equidistant and variable time steps. To this end, we leverage multistep and Taylor integrators. Exact inference has several advantages compared to approximations such as variational inference, e.g., it provides mathematical guarantees. In order to allow the usage of arbitrary integrators for predictions, we derive decoupled sampling schemes. This enables sampling consistent vector fields from the GP posterior. We show theoretically and empirically that integrators of sufficient order provide accurate ODE approximations. Interesting aspects for future work include the application of such schemes, e.g., to model-based reinforcement learning.

Acknowledgements

We thank Friedrich Solowjow for valuable discussions.

References

Abad, A.; Barrio, R.; Blesa, F.; and Rodríguez, M. 2012. Algorithm 924: TIDES, a Taylor Series Integrator for Differential Equations. *ACM Trans. Math. Softw.*, 39(1).

Brüdigam, J.; Schuck, M.; Capone, A.; Sosnowski, S.; and Hirche, S. 2022. Structure-Preserving Learning Using Gaussian Processes and Variational Integrators. In *Proceedings of The 4th Annual Learning for Dynamics and Control Conference*, volume 168 of *Proceedings of Machine Learning Research*, 1150–1162. PMLR.

Chen, R. T. Q.; Rubanova, Y.; Bettencourt, J.; and Duvenaud, D. K. 2018. Neural Ordinary Differential Equations. In *Advances in Neural Information Processing Systems 31*, 6571–6583.

Chowdhury, S. R.; and Gopalan, A. 2017. On Kernelized Multi-Armed Bandits. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, 844–853.

Cveticanin, L. 2013. On the Van Der Pol Oscillator: an Overview. In *Acoustics and Vibration of Mechanical Structures*, volume 430 of *Applied Mechanics and Materials*, 3–13.

Deisenroth, M. P.; and Rasmussen, C. E. 2011. PILCO: A Model-Based and Data-Efficient Approach to Policy Search. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, 465–472.

Djeumou, F.; Neary, C.; Goubault, E.; Putot, S.; and Topcu, U. 2022. Taylor-Lagrange Neural Ordinary Differential Equations: Toward Fast Training and Evaluation of Neural ODEs. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence*.

Doerr, A.; Daniel, C.; Schiegg, M.; Nguyen-Tuong, D.; Schaal, S.; Toussaint, M.; and Trimpe, S. 2018. Probabilistic Recurrent State-Space Models. In *Proceedings of the International Conference on Machine Learning (ICML)*.

Dondelinger, F.; Rogers, S.; and Husmeier, D. 2013. ODE parameter inference using adaptive gradient matching with Gaussian processes. In *Sixteenth International Conference on Artificial Intelligence and Statistics; AISTATS*.

Ensinger, K.; Solowjow, F.; Ziesche, S.; Tiemann, M.; and Trimpe, S. 2022. Structure-preserving Gaussian Process Dynamics. In *Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD 2022)*.

Fiedler, C.; Scherer, C. W.; and Trimpe, S. 2021. Practical and Rigorous Uncertainty Bounds for Gaussian Process Regression. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(8): 7439–7447.

Hairer, E.; Nørsett, S.; and Wanner, G. 1987. *Solving Ordinary Differential Equations I – Nonstiff Problems*. Springer.

Hegde, P.; Yıldız, c.; Lähdesmäki, H.; Kaski, S.; and Heinonen, M. 2022. Variational multiple shooting for Bayesian ODEs with Gaussian processes. In *Proceedings of the Thirty-Eighth Conference on Uncertainty in Artificial Intelligence*, volume 180 of *Proceedings of Machine Learning Research*, 790–799. PMLR.

Heinonen, M.; and d'Alché Buc, F. 2014. Learning nonparametric differential equations with operator-valued kernels and gradient matching. *arXiv preprint: arxiv.1411.5172*.

Heinonen, M.; Yildiz, C.; Mannerström, H.; Intosalmi, J.; and Lähdesmäki, H. 2018. Learning unknown ODE models with Gaussian processes. In *Proceedings of the 35th International Conference on Machine Learning*.

Hewing, L.; Arcari, E.; Fröhlich, L.; and Zeilinger, M. N. 2020. On Simulation and Trajectory Prediction with Gaussian Process Dynamics. In *Proceedings of the 2nd Conference on Learning for Dynamics and Control*, volume 120.

Howarth, R. J. 1979. Mining Geostatistics. London & New York (Academic Press), 1978. *Mineralogical Magazine*, 43: 1–4.

Keller, R. T.; and Du, Q. 2021. Discovery of Dynamics Using Linear Multistep Methods. *SIAM Journal on Numerical Analysis*, 59(1): 429–455.

Ljung, L. 1999. System identification. *Wiley encyclopedia of electrical and electronics engineering*, 1–19.

Mchutchon, A.; and Rasmussen, C. 2011. Gaussian Process Training with Input Noise. In *Advances in Neural Information Processing Systems*, volume 24.

Nguyen-Tuong, D.; and Peters, J. 2011. Model learning for robot control: A survey. *Cognitive processing*, 12: 319–40.

Ott, K.; Katiyar, P.; Hennig, P.; and Tiemann, M. 2021. ResNet After All: Neural ODEs and Their Numerical Solution. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*.

Rahimi, A.; and Recht, B. 2008. Random Features for Large-Scale Kernel Machines. In *Advances in Neural Information Processing Systems*, volume 20.

Raissi, M.; Perdikaris, P.; and Karniadakis, G. E. 2017. Numerical Gaussian Processes for Time-dependent and Non-linear Partial Differential Equations. *arXiv preprint: arXiv.2211.11103*.

Raissi, M.; Perdikaris, P.; and Karniadakis, G. E. 2018. Multistep Neural Networks for Data-driven Discovery of Nonlinear Dynamical Systems. *arXiv preprint: arXiv.1801.01236*.

Rasmussen, C. E.; and Williams, C. K. I. 2005. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press.

Rath, K.; Albert, C.; Bischl, B.; and Toussaint, U. 2021. Symplectic Gaussian process regression of maps in Hamiltonian systems. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 31: 053121.

Ridderbusch, S.; Ober-Blöbaum, S.; and Goulart, P. 2023. The past does matter: correlation of subsequent states in trajectory predictions of Gaussian Process models. In *Uncertainty in Artificial Intelligence, UAI 2023, July 31 - 4 August 2023, Pittsburgh, PA, USA*, volume 216 of *Proceedings of Machine Learning Research*, 1752–1761.

Schmidt, M.; and Lipson, H. 2009. Distilling Free-Form Natural Laws from Experimental Data. *Science*, 324(5923): 81–85.

Teymur, O.; Zygalakis, K.; and Calderhead, B. 2016. Probabilistic Linear Multistep Methods. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, 4321–4328.

Wenk, P.; Gotovos, A.; Bauer, S.; Gorbach, N.; Krause, A.; and Buhmann, J. M. 2019. Fast Gaussian Process Based Gradient Matching for Parameter Identification in Systems of Nonlinear ODEs. In *Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics (AISTATS)*, 1351–1360.

Wilson, J.; Borovitskiy, V.; Terenin, A.; Mostowsky, P.; and Deisenroth, M. 2020. Efficiently sampling functions from Gaussian process posteriors. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119, 10292–10302.

Zhu, A.; Jin, P.; Zhu, B.; and Tang, Y. 2022. On Numerical Integration in Neural Ordinary Differential Equations. In *Proceedings of the International Conference on Machine Learning*.

Zhu, A.; Wu, S.; and Tang, Y. 2022. Error analysis based on inverse modified differential equations for discovery of dynamics using linear multistep methods and deep learning. *arXiv preprint: arXiv.2209.12123*.