Latent GP-ODEs with Informative Priors

Anonymous Author(s)
Affiliation
Address
email

Abstract

For many complex systems the parametric form of the differential equation might be unknown or infeasible to determine. Earlier works have explored to model the unknown ODE system with a Gaussian Process model, however, the application has been limited to a low dimensional data setting. We propose a novel framework by combining a generative and a Bayesian nonparametric model. Our model learns a physically meaningful latent representation (position, momentum) and solves in the latent space an ODE system. The use of GP allows us to account for uncertainty as well as to extend our work with informative priors. We demonstrate our framework on an image rotation dataset. The method demonstrates its ability to learn dynamics from high dimensional data and we obtain state-of-the-art performance compared to earlier GP-based ODEs models on dynamic forecasting.

1 Introduction

One of the main properties of the physical world is that it evolves over time. This is often characterised in the form of an ordinary differential equation (ODE). A general ODE can be defined as:

$$\dot{\mathbf{x}}_t := \frac{d\mathbf{x}_t}{dt} = \mathbf{f}(\mathbf{x}_t) \tag{1}$$

with the evolution over time is specified as:

$$\mathbf{x}_T = \mathbf{x}_0 + \int_0^T \mathbf{f}(\mathbf{x}_t)dt \tag{2}$$

where the state of the system is captured by vector $\mathbf{x}_t \in \mathbb{R}^D$ and it evolves in time $t \in \mathbb{R}_+$ from an initial state $\mathbf{x}_0$ to a time point T following its time derivative $\mathbf{f}(\mathbf{x}_t)$. Traditionally the dynamics are defined by expert knowledge of the system given prior observations and the parameters are found by numerical optimization [2]. However, for many problems the functional (parametric) form of $\mathbf{f}$ might be unknown. Hence, in recent years there has been growing amount of research that tries to learn the underlying continuous-time dynamics from data [4, 17].

An increasingly popular research direction is to model the unknown ODE with a Gaussian Process (GP) prior [9, 18, 23]:

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

where we assume zero-mean GPs for simplicity and the covariance matrix $K(\mathbf{x}, \mathbf{x}')$ fully specifies the GP. In the remainder of this paper, we refer to ODE systems with GP priors as *GP-ODEs*.

Submitted to *A causal view on dynamical systems* workshop at NeurIPS 2022. Do not distribute.

The key property of GPs that make them applicable for learning unknown non-parametric ODE functions is that they encode functions where similar states $\mathbf{x}, \mathbf{x}'$ induce similar differentials $\mathbf{f}(\mathbf{x}), \mathbf{f}(\mathbf{x}')$. In particular, this similarity is captured by the kernel function $K(\mathbf{x}, \mathbf{x}')$, which also permits prior knowledge embedding into the model.

In order to perform the integration in (2), the differential function $\mathbf{f}$ must be continuous, whereas GPs are stochastic processes. This was first noticed by Heinonen et al. [9], who address the issue by discarding the GP uncertainty in order to obtain a continuous function. Recently, Wilson et al. [21] demonstrated that function-space sampling of GP posteriors is possible thanks to the Matheron's rule. Later, Hegde et al. [7], Yıldız et al. [23] showcased that this framework allows for sampling ODE instantiations, or vector fields $\mathbf{f}$, to simulate GP-ODE systems.

Despite achieving state-of-the-art results, GP-ODE models are demonstrated on *(i)* low-dimensional data settings, where the input is either some low dimension observations or simulation data of the actual ODE system, and *(ii)* without the use of prior knowledge of the system, which is an integral part of GP modeling. To the best of our knowledge, Solin et al. [18] is the only work in this line of research that considers a high-dimensional data problem by employing variational auto-encoders (VAEs) alongside with a GP-ODE system. Yet, they fail to train the feature encoder and dynamical model in an end-to-end fashion, leading to sub-optimal performance. This is in stark contrast with *static* VAE methods, which are proven to extract meaningful latent representations from high-dimensional data. Similarly, this line of research lacks a principle investigation of meaningful priors induced on GP-ODE systems, with an exception of [23] that modifies the ODE right-hand-side (2) to incorporate domain specific knowledge. Since the neural network based ODE models have been shown to benefit from inductive biases, such as second-order dynamics formulations [22], we believe GP-ODEs can also be improved by stronger priors.

In this work, we propose VAE-GP-ODE that extends previous work by learning dynamics from high-dimensional data with a structured GP prior. We propose a variational auto-encoder (VAE) to embed the high-dimensional input into a latent space. Subsequently, we model the latent space continuous-time dynamics with a Gaussian Process (GP). The use of GP allows us to specify structural priors over the model dynamics, hence, we are able to constrain our dynamics given prior knowledge of the system. Our model is trained end-to-end using variational inference. We showcase our models ability to learn, reproduce and forecast given a high-dimensional sequential input (image sequences).

2 Background

In this section, we review standard VAE and Sparse Gaussian process models.

2.1 Variational Auto-Encoders (VAEs)

The main task of an auto-encoder is to compress the input data $\mathbf{x} \in \mathbb{R}^D$ into a much lower dimensional $d \ll D$ latent variable $\mathbf{z} \in \mathbb{R}^d$ without loss of task-relevant information. Due to non-linear neural network likelihood mapping, $p_\theta(\mathbf{x}|\mathbf{z})$, the posterior in VAEs becomes intractable and hence is approximated by an amortized variational distribution $q_\phi(\mathbf{z}|\mathbf{x})$, where ϕ and θ are the parameters of the *encoders* and *decoders* [12, 16]. In a VAE setting the model parameters ϕ are estimated via variational inference where the optimization task is to maximize the evidence lower bound (ELBO) (4) or equivalently minimise the Kullback-Leibler divergence between the approximate and true probability distribution.

$$\log p(\mathbf{x}) \geq \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})}[\log p(\mathbf{x}|\mathbf{z})] - \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})}[\log q_\phi(\mathbf{z}|\mathbf{x}) - \log p(\mathbf{z})] \tag{4}$$

2.2 Sparse Gaussian Processes (SGPs)

Gaussian processes (GPs) (3) define priors over functions. In this work, we consider priors on vector-valued functions $\mathbf{f} : \mathbb{R}^D \mapsto \mathbb{R}^D$, which implies matrix-valued kernel functions $K(\mathbf{x}, \mathbf{x}') \in \mathbb{R}^{D\times D}$. While GPs allow for handling uncertainties, their computational complexity grows cubically with the number of input points. To tackle this, GPs are sparsified [14] by the introduction of inducing variables $\mathbf{U} = (\mathbf{u}_1, \ldots, \mathbf{u}_J)^T \in R^{J\times D}$ and inducing locations $\mathbf{M} = (\mathbf{m}_1, \ldots, \mathbf{m}_J)^T \in R^{J\times D}$

such that $\mathbf{u}_m = \mathbf{f}(\mathbf{m}_j)$. In turn, the function values $\mathbf{f}(\cdot)$ is evaluated conditioned on the inducing variables:

$$p(\mathbf{U}) = N(\mathbf{U}|\mathbf{0}, \mathbf{K_{MM}}) \tag{5}$$

$$p(\mathbf{f}|\mathbf{U}) = N(\mathbf{f}|\mathbf{A}\text{vec}(\mathbf{U}), \mathbf{K_{XX}} - \mathbf{A}\mathbf{K_{MM}}\mathbf{A^T}) \tag{6}$$

where $\mathbf{K}$ is the covariance matrix for the given input and $\mathbf{A} = \mathbf{K_{XM}}\mathbf{K_{MM}^{-1}}$. We have chosen to follow the SGP framework as it allows us to utilise mini batching during training and hence use stochastic gradient descent as an optimization algorithm.

3 Methods

We focus on problems where the goal is to learn an unknown ODE from a high-dimensional sequential input given prior knowledge of the system. In the following sections we first describe the generative and nonperametric Bayesian components of our model. Second, we define our optimisation objective following variational inference principles.

3.1 VAE-GP-ODE Model

In our work we follow the generative model specification as first introduced by Chen et al. [4]. In particular, our generative model involves latent variable(s) that correspond to the initial value(s) of the ODE system as well as a differential function to be inferred. Inspired by Yildiz et al. [22], we contrast first- and second-order ODE models, for which the corresponding latent variables are (a) the initial position, or (b) the initial position and initial velocity. As such the generative model can be expressed as follows (specifying for option (b), for option (a) ignore the velocity specification):

$$\mathbf{s}_0 \sim p(\mathbf{s}_0) \tag{7}$$

$$\mathbf{v}_0 \sim p(\mathbf{v}_0) \tag{8}$$

$$\mathbf{s}_t = \mathbf{s}_0 + \int_0^t \mathbf{v}_\tau d\tau \tag{9}$$

$$\mathbf{v}_t = \mathbf{v}_0 + \int_0^t \mathbf{f}(\mathbf{s}_\tau, \mathbf{v}_\tau) d\tau \tag{10}$$

$$\mathbf{x}_i \sim p(\mathbf{x}_i|\mathbf{s}_i) \quad i \in [0, N] \tag{11}$$

where the state position $\mathbf{s}_t$ and state velocity $\mathbf{v}_t$ together form the latent state vector $\mathbf{z}_t = (\mathbf{s}_t, \mathbf{v}_t)$. As it can be noted in Eq.11 the likelihood $p(\mathbf{x}|\mathbf{s})$ only depends on the state position while velocity acts as an auxiliary variable driving the dynamics.

GPs as Nonparametric ODE model In our work we consider both first and second-order ODEs. Second-order ODEs might be a favourable modelling choice as they are able to model higher order dynamics as well as they can act as an useful inductive bias leading to major performance improvements [6]. The first-order ODE follows Eq.1 while the second-order ODE has the following form:

$$\ddot{\mathbf{x}}_t := \frac{d^2\mathbf{x}_t}{dt^2} = \mathbf{f}(\boldsymbol{x}_t, \dot{\mathbf{x}}_t) \tag{12}$$

In the case of a second-order ODE we decouple the ODE into two first-order ODEs:

$$\begin{cases} \dot{\mathbf{s}}_t = \mathbf{v}_t \\ \dot{\mathbf{v}}_t = \mathbf{f}(\mathbf{s}_t, \mathbf{v}_t) \end{cases} \quad \begin{bmatrix} \mathbf{s}_T \\ \mathbf{v}_T \end{bmatrix} = \begin{bmatrix} \mathbf{s}_0 \\ \mathbf{v}_0 \end{bmatrix} + \int_0^T \begin{bmatrix} \mathbf{v}_t \\ \mathbf{f}(\mathbf{s}_t, \mathbf{v}_t) \end{bmatrix} dt \tag{13}$$

where we model $\mathbf{f}(\cdot)$ with a GP prior Eq.3. As the task at hand is to learn dynamics from high-dimensional input we follow the sparse variational inference framework for GPs using inducing variables [19].

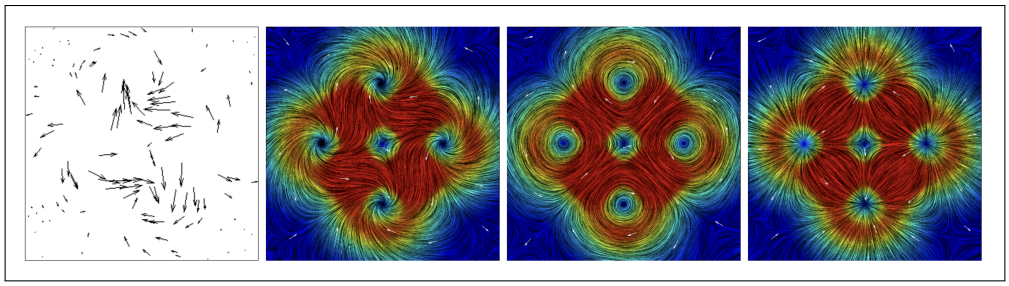


Figure 1: Learning a vector field. From lelt to right: samples, learned field, divergence- and curl-free components (kindly taken from [8]).

Discussion on informative priors for GPs The differential function $\mathbf{f}(\cdot) : \mathbb{R}^D \mapsto \mathbb{R}^D$ defines a vector field. Therefore, we model the vector field as a vector-valued GP [1]:

$$\mathbf{f} \sim GP(\mathbf{0}, \mathbf{K}) \tag{14}$$

where $\mathbf{K}$ is an operator valued kernel, meaning that each entry is a matrix valued kernel $K(\mathbf{x}, \mathbf{x}') \in \mathbb{R}^{D \times D}$ while the kernel matrix over data points $\mathbf{X} \in \mathbb{R}^{N \times D}$ becomes $\mathbf{K} = (K(\mathbf{x_i}, \mathbf{x_j}))_{i,j=1}^{N} \in \mathbb{R}^{ND \times ND}$. As a baseline implementation we consider the identity decomposable kernel with a radial basis function (rbf) scalar matrix:

$$K_{\text{rbf}}(\mathbf{x}, \mathbf{x}') = \sigma^2 \exp\left(-\frac{||\mathbf{x} - \mathbf{x}'||^2}{2l^2}\right) \tag{15}$$

where l and σ^2 are the kernel parameters, called lengthscale and variance, respectively. After which we consider more complex operator valued kernels, such as, divergence-free kernel that assumes stronger structural bias on the vector field:

$$K_{\text{df}}(\mathbf{x}, \mathbf{x}') = \frac{\sigma_f^2}{l^2} e^{-\frac{||\mathbf{x}-\mathbf{x}'||^2}{2l^2}} \left(\left(\frac{\mathbf{x} - \mathbf{x}'}{l}\right) \left(\frac{\mathbf{x} - \mathbf{x}'}{l}\right)^T + \left((D-1) - \frac{||\mathbf{x} - \mathbf{x}'||^2}{l^2}\right) \cdot \mathbf{I} \right) \tag{16}$$

where D is the number of input dimensions (for more details see the Appendix B). The resulting vector field does not have any sinks, therefore every state eventually returns to itself. In principle, one can construct a covariance function that captures the known physical characteristic of the underlying vector field (Fig. 1). The benefit of such an approach is that it reduces search space of the *true* model, as well as makes the predictions aligned with the known physical properties of a given system.

3.2 Variational Inference

The joint probability model Given the generative model (7-11), the model unknowns are the initial values $\mathbf{z}_0$, differential function $\mathbf{f}$, inducing inputs $\mathbf{Z}$ and inducing outputs $\mathbf{U}$. Overall, the full joint distribution becomes:

$$p(\mathbf{x}_{0:N}, \mathbf{f}, \mathbf{z}_0, \mathbf{U}, \mathbf{Z}) = \prod_{i=0}^{N} p(\mathbf{x}_i | \mathbf{f}, \mathbf{z}_0) p(\mathbf{z}_0) p(\mathbf{f}|\mathbf{U}) p(\mathbf{U}) p(\mathbf{Z}) \tag{17}$$

Due to non-linear likelihood mapping, the exact likelihood becomes intractable. Similar to previous work [4, 22], we resort to variational inference that involves amortized inference for the initial values and the mean-field inference framework for inducing variables [10].

Variational distributions Relying on the independence between initial values and dynamics we propose the following posterior factorization:

$$q(\mathbf{f}, \mathbf{U}, \mathbf{z}_0|\mathbf{x}_{0:N}) = q_{\text{enc}}(\mathbf{z}_0|\mathbf{x}_{0:N})q(\mathbf{f}, \mathbf{U}) \tag{18}$$
$$= q_{\text{enc}}(\mathbf{z}_0|\mathbf{x}_{0:N})p(\mathbf{f}|\mathbf{U})q(\mathbf{U}), \tag{19}$$

where the middle term was defined in (6). For initial values $\mathbf{z}_0$, we propose a *position encoder* for the first-order method while second-order formulation requires one *position encoder* and one *velocity encoder*. For the *position encoder*, the encoder takes as input only the first frame of the input sequence, whereas for the *velocity encoder* we consider the first n frames, where $n \ll N$ with N being the total sequence length. The variational encoding distribution is defined as

$$q_{\text{enc}}(\mathbf{z}_0|\mathbf{x}_{0:N}) = q_{\text{enc}}\left(\begin{pmatrix} s_0 \\ \mathbf{v}_0 \end{pmatrix} \middle| \mathbf{x}_{0:N}\right) = N\left(\begin{pmatrix} \mu_s(\mathbf{x}_0) \\ \mu_v(\mathbf{x}_{0:n}) \end{pmatrix}, \begin{pmatrix} \text{diag}(\sigma_s(\mathbf{x}_0)) & \mathbf{0} \\ \mathbf{0} & \text{diag}(\sigma_v(\mathbf{x}_{0:n})) \end{pmatrix}\right) \tag{20}$$

where $\mu_s, \mu_v, \sigma_s, \sigma_v$ are encoding neural networks. The variational approximation of the vector field is defined as

$$q(\mathbf{f}) = \int p(\mathbf{f}|\mathbf{U})q(\mathbf{U})d\mathbf{U} \tag{21}$$

where $q(\mathbf{U})$ is a factorised Gaussian of the inducing variables across state dimensions

$$q(\mathbf{U}) = \prod_{d=1}^{D} N(\mathbf{u}_d|\mathbf{m}_d, \mathbf{Q}_d) \tag{22}$$

with $\mathbf{u}_d \in \mathbb{R}^J$, $\mathbf{m}_d \in \mathbb{R}^J$ and $\mathbf{Q}_d \in \mathbb{R}^{J\times J}$. We finally note that the inducing locations $\mathbf{Z}$ are treated as kernel hyperparameters [10].

Evidence lower bound Our goal is to learn the vector field $\mathbf{f}$, the inducing points $\mathbf{U}$, as well as the parameters for the latent encoding $\mathbf{z}$. Hence, the evidence lower bound (ELBO) becomes as follows:

$$\log p(X) = \log \int\int\int p(X|\mathbf{z}_0, \mathbf{f})p(\mathbf{z}_0)p(\mathbf{f}|\mathbf{U})p(\mathbf{U})d\mathbf{z}_0 d\mathbf{f} d\mathbf{U} \tag{23}$$
$$= \log \int\int\int \frac{q(\mathbf{f}, \mathbf{U}, \mathbf{z}_0)}{q(\mathbf{f}, \mathbf{U}, \mathbf{z}_0)} p(X|\mathbf{z}_0, \mathbf{f})p(\mathbf{z}_0)p(\mathbf{f}|\mathbf{U})p(\mathbf{U})d\mathbf{z}_0 d\mathbf{f} d\mathbf{U} \tag{24}$$
$$\geq \underbrace{\mathbb{E}_{q(\mathbf{f},\mathbf{U},\mathbf{z_0})}\left[\log p(\mathbf{X}|\mathbf{z_0}, \mathbf{f})\right]}_{L_x} + \underbrace{\mathbb{E}_{q(\mathbf{z_0})}\left[\log \frac{p(\mathbf{z}_0)}{q_{\text{enc}}(\mathbf{z_0}|\mathbf{X})}\right]}_{L_z} + \underbrace{\mathbb{E}_{q(\mathbf{f},\mathbf{U})}\left[\log \frac{p(\mathbf{f}|\mathbf{U})p(\mathbf{U})}{p(\mathbf{f}|\mathbf{U})q(\mathbf{U})}\right]}_{L_u} \tag{25}$$

where L_x is the likelihood term, L_z is the regularization KL term and L_u inducing KL term (for details see Appendix A). The expected likelihood term L_x decomposes over time:

$$L_x = \sum_{i=1}^{N} \mathbb{E}_{q(\mathbf{f},\mathbf{U},\mathbf{z}_0)} \log p(\mathbf{x}_i|\mathbf{f}, \mathbf{z}_0). \tag{26}$$

We compute the likelihood $p(\mathbf{x}_i|\mathbf{f}, \mathbf{z}_0)$ over the ODE state solution $\mathbf{z}_i = \mathbf{z}_0 + \int_0^{t_i} \mathbf{f}(\mathbf{z}(\tau))d\tau$, which later is mapped to the data space via the decoder. Since the expectation is not available in closed form, we resort to Monte Carlo approximation, where we sample realizations of the vector field $\mathbf{f}^s \sim q(\mathbf{f}^s)$ and the encoder distributions $\mathbf{z}_0^s \sim q_{\text{enc}}(\mathbf{z}_0^s|\mathbf{x}_{0:m})$. Consequently, the likelihood term can be approximated as

$$L_x \approx \frac{1}{S}\sum_{s=1}^{S}\sum_{i=1}^{N} \log p(\mathbf{x}_i|\mathbf{f}^s, \mathbf{z}_0^s) = \frac{1}{S}\sum_{s=1}^{S}\sum_{i=1}^{N} \log p(\mathbf{x}_i|\text{dec}(\mathbf{z}_i^s)) \tag{27}$$

4 Experiments

We showcase the performance of our model on rotating MNIST dataset [3]. The first goal of our model is to showcase that *GP-ODE* type models can be used for large scale data, meaning that we are able to successfully predict the future frames of the sequence given the initial frame. The second goal is to constraint the dynamics given an informative prior allowing to extrapolate to time points outside the training domain.

The model was implemented in PyTorch. The encoder, Gaussian process and decoder parameters were all jointly optimized using the Adam optimizer [11] with learning rate 0.001. For the numerical solver we used the torchdiffeq package [5].

4.1 Rotating MNIST

For the dataset design we follow the experimental set-up as introduced by Casale et al. [3]. The dataset is created by rotating the images of handwritten "3" digits. In total we consider 16 rotation angels and 360 training sequences. We fix the data set to always have the same initial rotational angle. We set the dimensionality of the latent space to 16 and we keep the size of the latent space fixed for both first and second order ODE. We use a Bernoulli distribution for the reconstruction likelihood as in [22].

Both, first and second-order, VAE-GP-ODE model is able to perform the forward prediction given the initial frame with great accuracy, see Fig. 3a and Fig. 3b. The model has learnt both the dynamics as well as the style of a given digit. Visually inspecting the predicted images the performance is superior to earlier works [22, 18]. Test errors on the the reconstruction are given in table 1. Important to note, that the results were obtained without manual fine tuning of the kernel parameters or by adjusting the loss term, highlighting the methods robustness.

Table 1: Mean squared errors on test sequences (rbf kernel

VAE-GP-ODE	MSE (std)
first-order	0.009 ($\pm$0.034)
second-order	0.011 ($\pm$0.040)

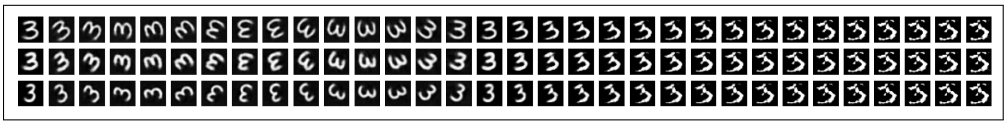
(a) First-order ODE

(b) Second-order ODE

Figure 2: Reconstructed test sequences. Top row ground truth, bottom row model prediction

On the other hand, neither one of the models is able to extrapolate to time points outside the training regime, $T > 16$, see Fig. 3a and Fig.3b. This observation comes in line with our latent space investigation, see Fig. 4 and Fig.5, where a circle indicates the initial state and the resulting sixteen latent states are plotted for each sample of the batch (25 samples). As no inductive bias is specified for the system, the current model does not learn to constraint the dynamics to angular rotation. Therefore, if the trajectory in latent space is forecasted beyond the training time points, the model is unable to *repeat* the rotation.

(a) First-order ODE



(b) Second-order ODE

Figure 3: Extrapolation to T=32

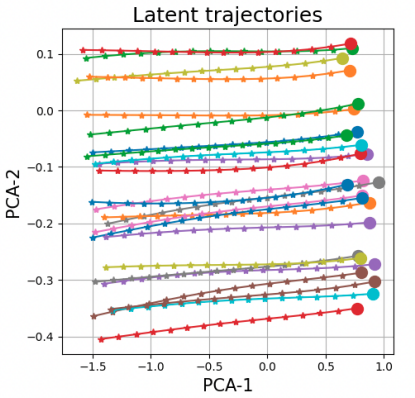


Figure 4: Latent space (first-order ODE)

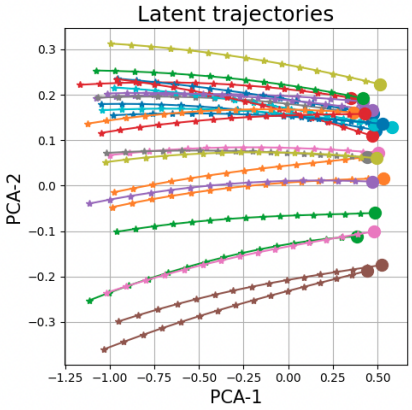


Figure 5: Latent space (second-order ODE)

In order to enforce the apparent rotational property of the dataset at hand we further investigate setting a GP with a divergence-free kernel specification. This should constrain the vector field of the latent space, every point should return at its initial location after certain time. Hence, the model should be able to extrapolate to time points beyond the training time.

The performance of the divergence-free kernel on forward predictions is on par with the rbf kernel. On visual inspection the reconstructed sequences exhibit the same level of accuracy, see Fig. 6, however the MSE on test sequences has slightly decreased 0.014 ($\pm$0.052). With respect to the extrapolation and latent space currently we do not see any major improvement from the divergence-free kernel as compared to the rbf kernel (see Appendix C). However, this is a preliminary implementation, therefore, the results are not yet conclusive and we are planning to investigate this further.

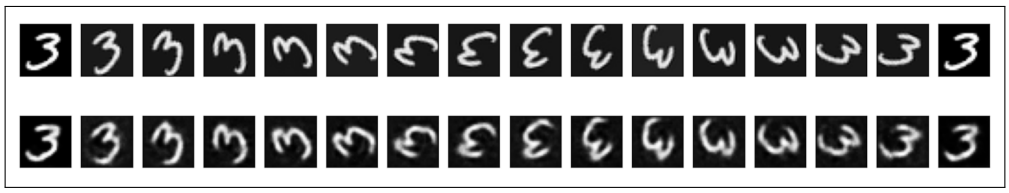


Figure 6: Reconstructed test sequences with DF-kernel (first-order ODE). Top row ground truth, bottom row model prediction

5 Discussion

In the present work we have introduced an extension to *GP-ODE* models for continuous-time dynamic modeling for high-dimensional data. We propose a generative model for feature extraction in combi-

nation with a nonparametric Bayesian model of the differential equation. As shown in preliminary results our framework obtains state-of-the-art forecasting performance on high-dimensional image sequences.

The present work is planned to be extended by informative priors that would constrain the viable dynamics. Furthermore, we assume that training in end-to-end fashion is advantageous as it ensures that the latent space learned by the generative model is relevant to dynamic modelling. In particular, in our work we assume that we either learn the state or the tuple of state and velocity. As such, we are interested in confirming our assumption by performing an ablation study where the generative model training is decoupled from the dynamic learning.

References

[1] M. A. Alvarez, L. Rosasco, N. D. Lawrence, et al. Kernels for vector-valued functions: A review. *Foundations and Trends® in Machine Learning*, 4(3):195–266, 2012.

[2] J. C. Butcher. *Numerical methods for ordinary differential equations*. John Wiley & Sons, 2016.

[3] F. P. Casale, A. Dalca, L. Saglietti, J. Listgarten, and N. Fusi. Gaussian process prior variational autoencoders. *Advances in neural information processing systems*, 31, 2018.

[4] R. T. Chen, Y. Rubanova, J. Bettencourt, and D. K. Duvenaud. Neural ordinary differential equations. *Advances in neural information processing systems*, 31, 2018.

[5] R. T. Q. Chen. torchdiffeq, 2018. URL `https://github.com/rtqichen/torchdiffeq`.

[6] N. Gruver, M. Finzi, S. Stanton, and A. G. Wilson. Deconstructing the inductive biases of hamiltonian neural networks. *arXiv preprint arXiv:2202.04836*, 2022.

[7] P. Hegde, Ç. Yıldız, H. Lähdesmäki, S. Kaski, and M. Heinonen. Bayesian inference of odes with gaussian processes. *arXiv preprint arXiv:2106.10905*, 2021.

[8] M. Heinonen. Learning with spectral kernels. University Lecture, 2017.

[9] M. Heinonen, C. Yildiz, H. Mannerström, J. Intosalmi, and H. Lähdesmäki. Learning unknown ode models with gaussian processes. In *International Conference on Machine Learning*, pages 1959–1968. PMLR, 2018.

[10] J. Hensman, N. Fusi, and N. D. Lawrence. Gaussian processes for big data. *arXiv preprint arXiv:1309.6835*, 2013.

[11] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[12] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[13] L. Milenska. *An Impact of Divergence-Free Magnetic Field Interpolation Using a Solenoidal Gaussian Process Kernel*. University of California, Santa Cruz, 2018.

[14] J. Quinonero-Candela and C. E. Rasmussen. A unifying view of sparse approximate gaussian process regression. *The Journal of Machine Learning Research*, 6:1939–1959, 2005.

[15] C. E. Rassmussen and C. K. Williams. *Gaussian processes for machine learning*. MIT press Cambridge, MA, 2006.

[16] D. J. Rezende, S. Mohamed, and D. Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *International conference on machine learning*, pages 1278–1286. PMLR, 2014.

[17] Y. Rubanova, R. T. Chen, and D. K. Duvenaud. Latent ordinary differential equations for irregularly-sampled time series. *Advances in neural information processing systems*, 32, 2019.

[18] A. Solin, E. Tamir, and P. Verma. Scalable inference in sdes by direct matching of the fokker–planck–kolmogorov equation. *Advances in Neural Information Processing Systems*, 34:417–429, 2021.

[19] M. Titsias. Variational learning of inducing variables in sparse gaussian processes. In *Artificial intelligence and statistics*, pages 567–574. PMLR, 2009.

[20] N. Wahlström. *Modeling of magnetic fields and extended objects for localization applications*. PhD thesis, Linköping University Electronic Press, 2015.

[21] J. Wilson, V. Borovitskiy, A. Terenin, P. Mostowsky, and M. Deisenroth. Efficiently sampling functions from gaussian process posteriors. In *International Conference on Machine Learning*, pages 10292–10302. PMLR, 2020.

[22] C. Yildiz, M. Heinonen, and H. Lahdesmaki. Ode2vae: Deep generative second order odes with bayesian neural networks. *Advances in Neural Information Processing Systems*, 32, 2019.

[23] Ç. Yıldız, M. Kandemir, and B. Rakitsch. Learning interacting dynamical systems with latent gaussian process odes. *arXiv preprint arXiv:2205.11894*, 2022.

A Detailed Derivations

ELBO With the above variational model specification we can derive the following evidence lower bound

$$\log p(X) = \log \int\int\int p(X|\mathbf{z}_0, \mathbf{f})p(\mathbf{z}_0)p(\mathbf{f}|\mathbf{U})p(\mathbf{U})d\mathbf{z}_0 d\mathbf{f} d\mathbf{U} \tag{28}$$

$$= \log \int\int\int \frac{q(\mathbf{f}, \mathbf{U}, \mathbf{z}_0)}{q(\mathbf{f}, \mathbf{U}, \mathbf{z}_0)} p(X|\mathbf{z}_0, \mathbf{f})p(\mathbf{z}_0)p(\mathbf{f}|\mathbf{U})p(\mathbf{U})d\mathbf{z}_0 d\mathbf{f} d\mathbf{U} \tag{29}$$

$$\geq \mathbb{E}_{q(\mathbf{f},\mathbf{U},\mathbf{z}_0)} \log \left[\frac{p(X|\mathbf{z}_0,\mathbf{f})p(\mathbf{z}_0)p(\mathbf{f}|\mathbf{U})p(\mathbf{U})}{q(\mathbf{z}_0|X)q(\mathbf{f},\mathbf{U})} \right] \tag{30}$$

$$\geq \mathbb{E}_{q(\mathbf{f},\mathbf{U},\mathbf{z}_0)} \left[\log p(X|\mathbf{z}_0, \mathbf{f}) + \log \frac{p(\mathbf{z}_0)}{q(\mathbf{z}_0|X)} + \log \frac{p(\mathbf{f}|\mathbf{U})p(\mathbf{U})}{p(\mathbf{f}|\mathbf{U})q(\mathbf{U})} \right] \tag{31}$$

$$\geq \underbrace{\mathbb{E}_{q(\mathbf{f},\mathbf{U},\mathbf{z}_0)} \left[\log p(X|\mathbf{z}_0, \mathbf{f}) \right]}_{L_x} + \underbrace{\mathbb{E}_{q(\mathbf{f},\mathbf{U},\mathbf{Z})} \left[\log \frac{p(\mathbf{z}_0)}{q(\mathbf{z}_0|X)} \right]}_{L_z} + \underbrace{\mathbb{E}_{q(\mathbf{f},\mathbf{U},\mathbf{Z})} \left[\log \frac{p(\mathbf{f}|\mathbf{U})p(\mathbf{U})}{p(\mathbf{f}|\mathbf{U})q(\mathbf{U})} \right]}_{L_u} \tag{32}$$

Hence the ELBO decomposes as

$$L = L_x + L_z + L_u \tag{33}$$

Regularization Term The regularization term L_z corresponds to KL divergence between the encoding distribution and the prior on the latent space.

$$L_z = \mathbb{E}_{q(\mathbf{f},\mathbf{U},\mathbf{z}_0)} \log \frac{p(\mathbf{z}_0)}{q(\mathbf{z}_0|X)} \tag{34}$$

$$= -KL\left[q_{\text{enc}}(\mathbf{z}_0|\mathbf{x}_{0:N})||p(\mathbf{z}_0)\right] \tag{35}$$

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

$$L_u = \mathbb{E}_{q(\mathbf{f},\mathbf{U},\mathbf{z}_0)} \log \frac{p(\mathbf{f}|\mathbf{U})p(\mathbf{U})}{p(\mathbf{f}|\mathbf{U})q(\mathbf{U})} \tag{36}$$

$$= -KL\left[q(\mathbf{U})||p(\mathbf{U})\right] \tag{37}$$

B Kernel Specifications

Divergence-free kernel In the same way as Gaussian distributions are closed under linear transformations, Gaussian Processes are closed under linear operators [15]. An example of such linear operator could be differentiation or integration (for further details see [20]). Using this property we can construct new covariance functions. One of such functions is a divergence-free covariance function.

In particular, a vector field $\mathbf{g}(\mathbf{u})$ is known to be solenoidal if it can written as the curl of another vector field $\mathbf{f}(\mathbf{u})$

$$\mathbf{g}(\mathbf{u}) = \nabla \times \mathbf{f}(\mathbf{u}) \tag{38}$$

We model the vector field $\mathbf{f}(\mathbf{u})$ with a GP, where each component $f_i(\mathbf{u})$ is a GP with a squared exponential covariance function (RBF kernel)

$$f(\mathbf{u}) \sim GP(\mathbf{0}, k_{\text{rbf}}(\mathbf{u}, \mathbf{u}')) \tag{39}$$

The resulting covariance function of $\mathbf{g}(\mathbf{u})$ is given by

$$K(x, x') = \frac{\sigma_f^2}{l^2} e^{-\frac{||\mathbf{x}-\mathbf{x}'||^2}{2l^2}} \left(\left(\frac{\mathbf{x}-\mathbf{x}'}{l} \right) \left(\frac{\mathbf{x}-\mathbf{x}'}{l} \right)^T + \left((n-1) - \frac{||\mathbf{x}-\mathbf{x}'||^2}{l^2} \right) \cdot \mathbf{I} \right) \tag{40}$$

where n is the dimension of the physical space. For a complete derivation see Wahlström [20] Appendix A. Important note on the above kernel is that it is only divergence-free analytically, in fact [13] has shown that the numerical divergence is non-zero.

C Supplementary Results Divergence-Free Kernel

Figure 7: Extrapolation to T=32 with DF-kernel (first-order ODE)

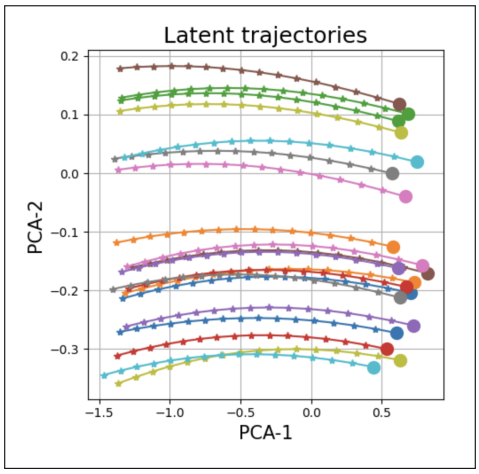


Figure 8: Latent space (first-order ODE) with DF-kernel