# Variational Inference for Composite Gaussian Process Models

**Jakob Lindinger**

Universitätsdissertation
zur Erlangung des akademischen Grades

Doktor der Ingenieurwissenschaften
*(Dr.-Ing.)*

in der Wissenschaftsdisziplin
Digital Health - Machine Learning



Angefertigt am

Fachgebiet Digital Health - Machine Learning
und am
Bosch Center for Artificial Intelligence

eingereicht an der

Digital-Engineering-Fakultät
der Universität Potsdam

**Datum der Disputation:** 17. Juli 2023

Betreuer
**Prof. Dr. Christoph Lippert**
Hasso Plattner Institute, University of Potsdam

Gutachter
**Prof. Dr. Marc Deisenroth**
University College London

**Prof. Dr. Ralf Herbrich**
Hasso Plattner Institute, University of Potsdam

Published online on the
Publication Server of the University of Potsdam:
https://doi.org/10.25932/publishup-60444
https://nbn-resolving.org/urn:nbn:de:kobv:517-opus4-604441

# Abstract

Most machine learning methods provide only point estimates when being queried to predict on new data. This is problematic when the data is corrupted by noise, e.g. from imperfect measurements, or when the queried data point is very different to the data that the machine learning model has been trained with. Probabilistic modelling in machine learning naturally equips predictions with corresponding uncertainty estimates which allows a practitioner to incorporate information about measurement noise into the modelling process and to know when not to trust the predictions. A well-understood, flexible probabilistic framework is provided by Gaussian processes that are ideal as building blocks of probabilistic models. They lend themself naturally to the problem of regression, i.e., being given a set of inputs and corresponding observations and then predicting likely observations for new unseen inputs, and can also be adapted to many more machine learning tasks. However, exactly inferring the optimal parameters of such a Gaussian process model (in a computationally tractable manner) is only possible for regression tasks in small data regimes. Otherwise, approximate inference methods are needed, the most prominent of which is variational inference.

In this dissertation we study models that are composed of Gaussian processes embedded in other models in order to make those more flexible and/or probabilistic. The first example are deep Gaussian processes which can be thought of as a small network of Gaussian processes and which can be employed for flexible regression. The second model class that we study are Gaussian process state-space models. These can be used for time-series modelling, i.e., the task of being given a stream of data ordered by time and then predicting future observations. For both model classes the state-of-the-art approaches offer a trade-off between expressive models and computational properties (e.g. speed or convergence properties) and mostly employ variational inference. Our goal is to improve inference in both models by first getting a deep understanding of the existing methods and then, based on this, to design better inference methods. We achieve this by either exploring the existing trade-offs or by providing general improvements applicable to multiple methods.

We first provide an extensive background, introducing Gaussian processes and their sparse (approximate and efficient) variants. We continue with a description of the models under consideration in this thesis, deep Gaussian processes and Gaussian process state-space models, including detailed derivations and a theoretical comparison of existing methods.

Then we start analysing deep Gaussian processes more closely: Trading off the properties (good optimisation versus expressivity) of state-of-the-art methods in this field, we propose a new variational inference based approach. We then demonstrate experimentally that our new algorithm leads to better calibrated uncertainty estimates than existing methods.

Next, we turn our attention to Gaussian process state-space models, where we closely analyse the theoretical properties of existing methods. The understanding gained in this process leads us to propose a new inference scheme for general Gaussian process state-space models that incorporates effects on multiple time scales. This method is more efficient than previous approaches for long time-series and outperforms its comparison partners on data sets in which effects on multiple time scales (fast and slowly varying dynamics) are present.

Finally, we propose a new inference approach for Gaussian process state-space models that trades off the properties of state-of-the-art methods in this field. By combining variational inference with another approximate inference method, the Laplace approximation, we design an efficient algorithm that outperforms its comparison partners since it achieves better calibrated uncertainties.

# Acknowledgements

I gratefully acknowledge the support of my university supervisor, Christoph. I am thankful for regular and insightful discussions which allowed me to place my work in a broader context and see helpful similarities. My Bosch supervisor, Barbara, and I both greatly appreciate the freedom Christoph provided in choosing and shaping the research questions that we followed and which led to this thesis. His experience in "selling" machine learning research was invaluable close to paper, poster, or talk deadlines.

Furthermore, I want to sincerely thank my Bosch supervisor, Barbara, for her priceless advice, explanations, ideas and her help with some of the experiments and during the (often hectic) paper writing phases. I have learned a great deal about research and how to perform proper machine learning experiments from Barbara, and I imagine I could especially learn quite a bit more about coming up with valuable and meaningful research questions from her. Discussions with her were always fruitful also thanks to her calm, helpful and friendly manner, and I benefited especially from her patience when helping me transform my first (mostly extremely technical) drafts of papers, posters or talks into a human-understandable form. I am immensely grateful for the time Barbara invested in supervising me which she managed to allocate despite her many other duties and research projects. Without her effort and guidance this thesis would not have been possible.

Even though there was a pandemic going on through most of the time I worked on this thesis, I never had to feel isolated due to the many people I had the pleasure to work with: Thanks for many interesting research-related (and also many more non-research related) discussions to my fellow PhD students (including but not limited to Manuel, the Katharinas, Alex, and Çağatay), the members of my Bosch research group (especially David, Sebastian, Jan, and Buote) and to the members of Christoph's research group that openly welcomed me the few times I visited in person. I am also grateful to the countless people that participated actively in one of the organising teams of the Bosch PhD program and thus helped to create a unique and extremely broad PhD experience.

Doing an industry PhD entails diverse experiences, but also leads to some organisational overhead. I am extremely grateful to Corinna, Lena, and Kevin for the organisational help with topics concerning my Bosch research group (and Bosch as a whole), concerning the HPI and the university of Potsdam, and concerning particularities of the Bosch PhD program, respectively.

Luckily, the time needed for finishing a PhD is not solely filled with work but also with much needed diversions, fun activities or other every-day things that are needed for a healthy work-life balance. My family and friends always took me in when I escaped from Swabia and provided me with all of the above and usually more. Whenever I needed some exercise to balance the work days sitting in the (home) office, I could rely on members of the Ultimate Frisbee or Boulder groups, which was always lots of fun.

Finally, I want to thank the most important person in my life, my partner Sabrina: For cheering me up when things were not going as I wished them to, for (possibly slightly obsessed) Gloomhaven sessions during and outside of the Corona lock-downs, and much needed other diversions from work. Without you the time in Renningen would not have been half as much fun. Thank you for embarking on the complete PhD journey together with me (without actually having signed up for it). Without you and your constant moral support this thesis would not have been possible.

# Contents

CHAPTER 1

# Introduction

## 1.1 Gaussian Process Regression

The use of Gaussian processes for regression can be independently found in many different fields such as geostatistics (e.g. Matheron, 1973; Journel and Huijbregts, 2003), meteorology (e.g. Thompson, 1956; Daley, 1993), and spatial prediction and spatial statistics (Whittle, 1963; Ripley, 1981). See also Rasmussen and Williams (2006, Sec. 2.8) for a more detailed history. Gradually it has been realised that Gaussian processes (GPs) can be used for generic regression problems (O'Hagan, 1978), with applications to e.g. computer experiments and their design (Sacks et al., 1989). With the seminal finding by Neal (1996) that infinitely wide neural networks correspond to GPs, the attention of the machine learning community was directed towards GPs. Their use in the regression context was quickly adopted (Williams and Rasmussen, 1995). In the following we introduce some basic background on GP regression before discussing specific applications.

A GP is a distribution over functions and completely specified by a mean function and a kernel (Rasmussen and Williams, 2006). Throughout this thesis we work with a mean function being constantly zero unless otherwise specified.[1] We denote a function $f : \mathbb{R}^d \to \mathbb{R}$ that is distributed according to a zero-mean GP with kernel $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ as $f \sim \mathcal{GP}(0, k)$, where $d$ is the dimensionality of the input space. Of practical relevance is the property of GPs that for every finite set with $N \in \mathbb{N}$ input points $X_N = \{x_n\}_{n=1}^N$ with $x_n \in \mathbb{R}^d$, the distribution over the function values $F_N \equiv \{f(x_n)\}_{n=1}^N \in \mathbb{R}^N$ is given as

$$p(F_N) = \mathcal{N}(F_N|0, K_{NN}) . \tag{1.1}$$

Here $K_{NN} \in \mathbb{R}^{N\times N}$ is the matrix obtained by evaluating the kernel at all pairs of input points, i.e., $K_{NN} \equiv \{k(x_n, x_{n'})\}_{n,n'=1}^N$ and we use $\mathcal{N}(x|\mu, \Sigma)$ to denote a multivariate Gaussian distribution (see e.g. Bishop, 2006, Sec. 2.3) over a (multi-dimensional) random variable $x$ with mean vector $\mu$ and covariance matrix $\Sigma$. We depict samples from Eq. (1.1) for different choices of kernels $k$ in Fig. 1.1 (top).

Regression is the problem of being given a set of input and corresponding output points $\{X_N, Y_N\}$ (the training data), where $Y_N = \{y_n\}_{n=1}^N$ with $y_n \in \mathbb{R}$, from which the output $y_*$ at an unseen input point $x_*$ has to be predicted. The Bayesian approach to regression (see e.g. Bishop, 2006, Sec. 1.2; van der Wilk, 2019, Chap. 1) is to place a so-called *prior* on the function describing the mapping from inputs to outputs. This allows to incorporate prior knowledge about the expected mapping (e.g. smoothness or periodicity) in the model, protects to some extent against overfitting to the training data and is rewarded by probabilistic predictions, i.e., a distribution over test outputs $p(y_*)$. In GP regression, we assume that the mapping can be described by a function $f$ with $y_n = f(x_n)$ and that this function can be described by a GP.[2] The prior on this functional mapping is then completely described by Eq. (1.1) and, through $K_{NN}$, depends on the chosen kernel $k$. By choosing different

[1] See e.g. Rasmussen and Williams (2006, Sec. 2.7) for a detailed discussion about mean functions of GPs.

[2] Typically the function $f$ is assumed to be unobserved and corrupted by additive observation noise. We discuss this in Sec. 2.1

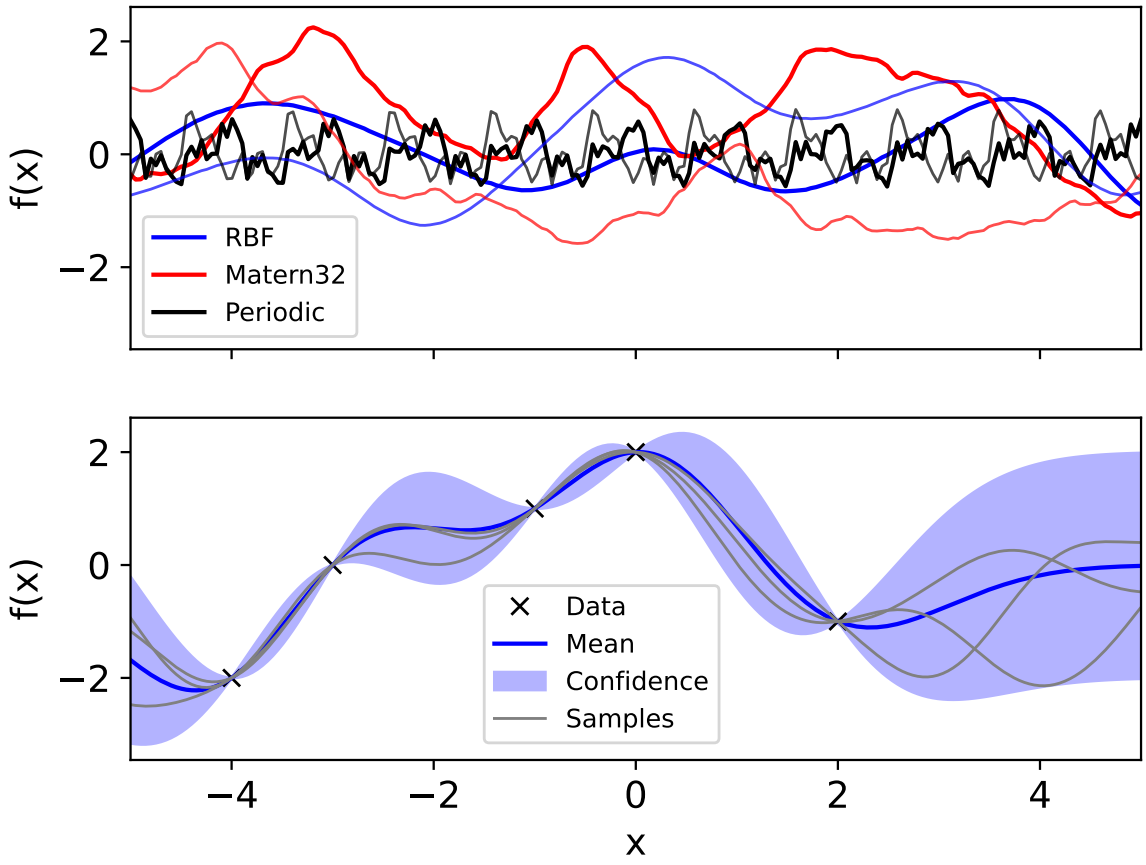


**Figure 1.1:** Top: Shown are two samples from the prior Eq. (1.1) for three different choices of the underlying kernel $k$, respectively, where the inputs $X_N$ are linearly spaced in the range $[-5, 5]$. The chosen kernels possess different properties, e.g. the radial basis function (RBF, blue) and the Matern-$3/2$ kernel (red) produce infinitely often and twice differentiable functions, respectively. The periodic kernel (black) leads to periodic functions, here with a chosen period of 1. Bottom: After having seen data (black crosses), we update our prior belief, leading to the posterior depicted here. We have chosen the RBF kernel and show the mean of the posterior prediction (solid blue line), along with a confidence interval (2 standard deviations, shaded blue area). The grey lines show different sample functions drawn from the posterior.

kernels (see e.g. Rasmussen and Williams, 2006, Chap. 4; Duvenaud, 2014, Chap. 2), we can express our prior belief about different properties that we expect the function $f$ to possess, such as smoothness, differentiability, stationarity, or periodicity (see Fig. 1.1, top).

Using the observations $Y_N$, we can update our belief about the underlying functional mapping, resulting in the so-called *posterior* $p(F_N|Y_N)$. The latter allows us to calculate the desired predictive distribution over the outputs $p(y_*|Y_N)$ which is depicted for a simple problem in Fig. 1.1 (bottom). We defer the formulas for these distributions and their derivations to Sec. 2.1. Importantly, the calculation of these distributions is analytically possible without having to rely on approximations, but incurs an $\mathcal{O}(N^3)$ cost since the matrix $K_{NN}$ appearing in Eq. (1.1) has to be inverted.

Gaussian process regression is optimally suited for cases where a function that is expensive to evaluate is to be approximated by as little observations as possible. The probabilistic nature prevents overfitting in these cases and the uncertainty predictions help reasoning about observations at unseen test locations. A very prominent example of such an application is Bayesian optimisation (e.g. Srinivas et al., 2010; Snoek et al., 2012; Shahriari et al., 2016; Eriksson et al., 2019) where the goal is to find the optimum of an expensive black-box function $f$ in as little evaluations as possible. Further

examples include active learning (sometimes also called optimal experimental design, see e.g. Krause et al., 2008; Settles, 2009; Schreiter et al., 2015), where the goal is to explore an unknown function $f$ with as little queries as possible, or Bayesian quadrature (e.g. O'Hagan, 1991; Gunter et al., 2014; Briol et al., 2019), where the goal is to approximate an integral over an expensive function $f$ with as little function evaluations as possible. The above-mentioned examples have in common that a GP can be naturally employed to model the unknown (and usually expensive) function $f$ which is done via standard GP regression as introduced above. However, for different tasks in machine learning, GP regression is simply not suitable or the formulation introduced so far is suboptimal (e.g. due to the computational burden). In the following we describe how to adapt GPs in order to tackle different challenges.

## 1.2 Challenges and Extensions for Gaussian Process Regression

While regression is the field of Machine Learning to which Gaussian processes can be most naturally applied to, their success led to the exploration of different fields: One example is classification, i.e., the task of mapping an input $x$ to one of $C$ classes $\mathcal{C}_1, \dots, \mathcal{C}_C$ (e.g. Williams and Barber, 1998; Rasmussen and Williams, 2006, Chap. 3). Additionally, other works were concerned with improvements for GP regression in specific settings, e.g. when the regression data arrives sequentially and the model has to be updated often (so-called online GPs, see e.g. Csató and Opper, 2002), or when the dimensionality of the input data is large and has to be reduced to a manageable size first (so-called manifold GPs, see e.g. Snelson and Ghahramani, 2006; Calandra et al., 2016). General dimensionality reduction techniques using GPs (Lawrence, 2005) or applications to regression with multiple correlated outputs also exist (multi-output GPs, e.g. Bonilla et al., 2007; Álvarez et al., 2012).

In the following, we focus on three particular challenges and extensions for GP regression: First, in Sec. 1.2.1, we briefly discuss sparse GPs that are required to make GP regression applicable to large data sets. Then, we discuss reasons for and ways towards making GP regression more flexible and expressive (Sec. 1.2.2). Finally, in Sec. 1.2.3, we discuss the applicability of GPs to time-series modelling, which is a standard problem in statistics and machine learning.

### 1.2.1 Sparse Gaussian Processes

The same problem as in online GPs, namely the $\mathcal{O}(N^3)$ scaling of GPs with the number of data points $N$, is tackled in a more general way by so-called *sparse GPs* (see e.g. Liu et al., 2020 for a recent review). Their goal is to find a way to lower the computational complexity of GPs while still providing good approximations to the posterior $p(F_N|Y_N)$. The predominant approach relies on introducing a set of pseudo-data $\{X_M, F_M\}$, where $X_M = \{x_m\}_{m=1}^M$ with $x_m \in \mathbb{R}^d$ are the so-called *inducing inputs* and $F_M \equiv \{f(x_m)\}_{m=1}^M$ the so-called *inducing outputs*. Their aim is to summarise the training data and by choosing $M \ll N$ to provide significant computational savings (Quinonero-Candela and Rasmussen, 2005). Over the recent years methods that additionally use an approximate inference scheme called *variational inference* (Blei et al., 2017) that aims to approximate the posterior $p(F_N, F_M|Y_N)$ by a parametric distribution $q(F_N, F_M)$ have become the most prominent sparse GP approach. We provide the technical details about these sparse variational GPs and discuss other approaches to sparse GPs in Sec. 2.2.

Advances in sparse GPs have typically been picked up quite quickly in other methods relying on GPs in order to reduce the computational complexity. This includes many of the examples mentioned above such as Bayesian optimisation (McIntire et al., 2016), multi-output GPs (Álvarez and Lawrence, 2011; Nguyen and Bonilla, 2014), or online GPs (Cheng and Boots, 2016; Bui et al., 2017). Many more examples of applications of sparse GPs can be found in the recent review by Liu et al. (2020).

### 1.2.2 Flexible and Expressive Regression

Another challenge for GP regression is the choice of the kernel $k$.[3] The example in Fig. 1.1 (bottom) did not really provide such a challenge due to its simplicity. A more instructive example is provided in Sec. 5.4.3 of Rasmussen and Williams (2006), where the concentration of $CO_2$ in the atmosphere for the years 1958 to 2003 at Mauna Loa is to be modelled and predicted into the future. To this end, a combination of multiple kernels (called kernel engineering, which yields again a valid kernel) describing seasonal and long-term trends as well as small deviations is constructed which is able to describe the data very well. While it is feasible for this example, the design of these kind of kernels typically requires expert knowledge in the application area mixed with a proficiency for kernels themselves.[4] Additionally, kernel engineering involves the risk of overfitting if too many or the wrong kernel components are chosen. The same two problems exist for feature engineering, where an expert proposes a mapping of the inputs (features) $X_N \to X'_N$ such that the new features $X'_N$ more closely resemble the physical quantities which are necessary for describing the mapping $X'_N \to Y_N$. A practitioner therefore requires a method that is flexible and expressive enough without the need of having to perform feature or kernel engineering first.

The first step in this direction was to introduce a learnable warping of either the outputs (Snelson et al., 2003) or the inputs (e.g. Snoek et al., 2014), which, in combination with standard kernels, is powerful enough to model difficult regression problems. Other approaches used a combination of two GPs in order to create more powerful models, e.g. by using the first GP to map the inputs to a latent space on which the second GP performs standard GP regression (Schmidt and O'Hagan, 2003; Damianou et al., 2011) or by defining a product model from the two GPs (Adams and Stegle, 2008). With the advent of deep learning (Krizhevsky et al., 2012; Goodfellow et al., 2016) came also the idea to combine GPs with neural networks. One approach, deep kernel learning, is using a deep neural network to warp the input for GPs (Wilson et al., 2016; Calandra et al., 2016), which has recently been shown to be susceptible to overfitting due to the large number of additional parameters of the neural network that are added to the model (Ober et al., 2021). A different method uses so-called *deep GPs*, extending the idea of stacking two GPs to using a small network of GPs arranged in layers in order to learn probabilistic representations of the data that can finally be treated as in standard GP regression (Damianou and Lawrence, 2013). This model class will be the focus of Chap. 3 of this thesis and we will discuss state-of-the-art approaches in Sec. 1.3.

### 1.2.3 Time Series Modelling

The last extension for GP regression that we wish to discuss here is time series modelling. In this problem, the goal is to predict future observations from an observed time series $Y_T = \{y_t\}_{t=1}^T$, where $y_t \in \mathbb{R}$ and $t$ is the time index marking a time $t\Delta_t$ after the intital time $t = 0$ and $\Delta_t$ is a constant time increment.[5] While it is possible for simple examples to interpret time series modelling as a regression problem (by interpreting the time index as the input and the observation as the output) as is done for the Mauna Loa problem mentioned at the beginning of Sec. 1.2.2, the important role that the time is playing is completely ignored in such an approach. More powerful methods respect the natural ordering of time and exploit this fact by considering the history to predict the present (and the future): One example are non-linear auto-regressive approaches that model the current observation as $y_t = f(y_{t-1}, \dots, y_{t-n_L})$, where $n_L$ is the lag, i.e., the number of past observations that is taken into account. We can simply make this a probabilistic method by adding Gaussian noise and placing a Gaussian process prior on the function $f$ (see e.g. Kocijan et al., 2005 or Mattos et al., 2016 and

[3] The parameters of the kernel, the so-called hyperparameters, have to be inferred as well. We discuss this in Sec. 2.1

[4] Automatic methods for performing kernel engineering also exist, see e.g. Duvenaud et al. (2013) or Bitzer et al. (2022).

[5] Throughout this thesis we assume one-dimensional time series with constant time increments $\Delta_t$ unless mentioned otherwise.

references therein). In such a system the propagation of the uncertainty coming from the transition from one time step to the next is challenging (see e.g. Girard et al., 2003).

An approach that naturally incorporates the latter uncertainty considers the time series modelling problem as one where the $y_t$ are interpreted as the observations of the state of a system $x_t$ that changes depending on this state and its history. A classical probabilistic approach that uses this interpretation is the so-called state-space model (see e.g. Särkkä, 2013) that aims to separate the observations $Y_T$ from observation noise through temporal latent (unobserved) states $X_T = \{x_t\}_{t=1}^T$, where $x_t \in \mathbb{R}^d$. The latent states and the observations are connected through an *emission model* $p(y_t|x_t)$ and the latent states are assumed to evolve according to a Markovian[6] *transition model* $p(x_t|x_{t-1})$. For the simplest case, where transition and emission model are linear with Gaussian noise, i.e., $p(y_t|x_t) = \mathcal{N}\left(y_t \middle| Ax_t + a, \sigma_y^2\right)$ and $p(x_t|x_{t-1}) = \mathcal{N}\left(x_t \middle| Bx_{t+1} + b, \sigma_x^2\right)$ (with real parameters $A, a, B, b, \sigma_x, \sigma_y$ of appropriate dimensionality) this formulation is analytically tractable as was shown in a seminal work by Kalman (1960). However, real-world problems are typically too complicated to be described by such a simple approach, instead requiring more expressive transition or emission models. The Bayesian non-parametric solution to this problem is to use an expressive transition model $p(x_t|x_{t-1}, f) = \mathcal{N}\left(x_t \middle| f(x_{t-1}), \sigma_x^2\right)$ and to place a GP prior on the function $f$, thus arriving at an approach called Gaussian process state-space model (GPSSM, Wang et al., 2005; Frigola, 2015). This model class will be the focus of Chaps. 4, 5 and 6 of this thesis and we will discuss state-of-the-art approaches in Sec. 1.4.

## 1.3 Introduction to Deep Gaussian Processes

As we introduced above in Sec. 1.2.2, deep GPs aim at performing flexible regression by chaining multiple GPs with standard kernels in a small network with $L$ layers. This leads to the regression assumption $y_n = f^L(\cdots f^1(x_n))$, where $f^l$ is the function given by the GP mapping in the $l$-th layer. In its simplest form this amounts to two layers with a single GP each, which is already sufficient to describe data with varying length scales which is difficult with standard kernels. This is schematically depicted in Fig. 1.2.

In order to describe more complex mappings, we can use larger networks of GPs with $L$ layers and $T_l$ GPs in the $l$-th layer (see Fig. 1.3 for a schematic depiction). However, the added flexibility of deep GPs comes at the price of losing the analytical tractability of standard GPs: The output of one layer is a distribution which can no longer be analytically fed through the next layer of GPs since the kernels through which the inputs are processed cannot deal analytically with distributions. Therefore, inference in deep GPs has to be performed approximately for which the current state-of-the-art methods (Salimbeni and Deisenroth, 2017; Havasi et al., 2018) rely on sparse GPs which also allows these approaches to scale to large data sets. As explained in Sec. 1.2.1, sparse GPs rely on pseudo data to summarise the training data. In the case of deep GPs this has to be done for every GP in the model, leading to the introduction of $\{X_M, F_M\}$, where $F_M = \{F_M^{l,t}\}_{t=1,l=1}^{T_l,L}$, where $F_M^{l,t}$ summarises the outputs $F_N^{l,t}$ of GP $f^{l,t}$, i.e., the $t$-th GP in layer $l$ (see Fig. 1.3).[7] Inference in deep GPs therefore has to find a way to approximate the posterior $p(F_N, F_M|Y_N)$, where $F_N = \{F_N^{l,t}\}_{t=1,l=1}^{T_l,L}$.

The method by Salimbeni and Deisenroth (2017) employs stochastic variational inference: It assumes independent multivariate Gaussian distributions $q(F_M^{l,t})$ over the inducing outputs $F_M^{l,t}$ which

[6] Here, Markovian means that every latent state is assumed to capture the complete history of latent states before, i.e., only one previous state is needed to describe the current state. Mathematically this can be expressed as $p(x_t|x_{t-1}, \ldots, x_1) = p(x_t|x_{t-1})$.

[7] With a slight abuse of notation we always use $F_M$ to denote the whole set of inducing outputs which leads to an inconsistent meaning when switching from sparse to deep GPs. For the sake of an uncluttered notation we accept this inconsistency since the meaning of $F_M$ is clear from the context.

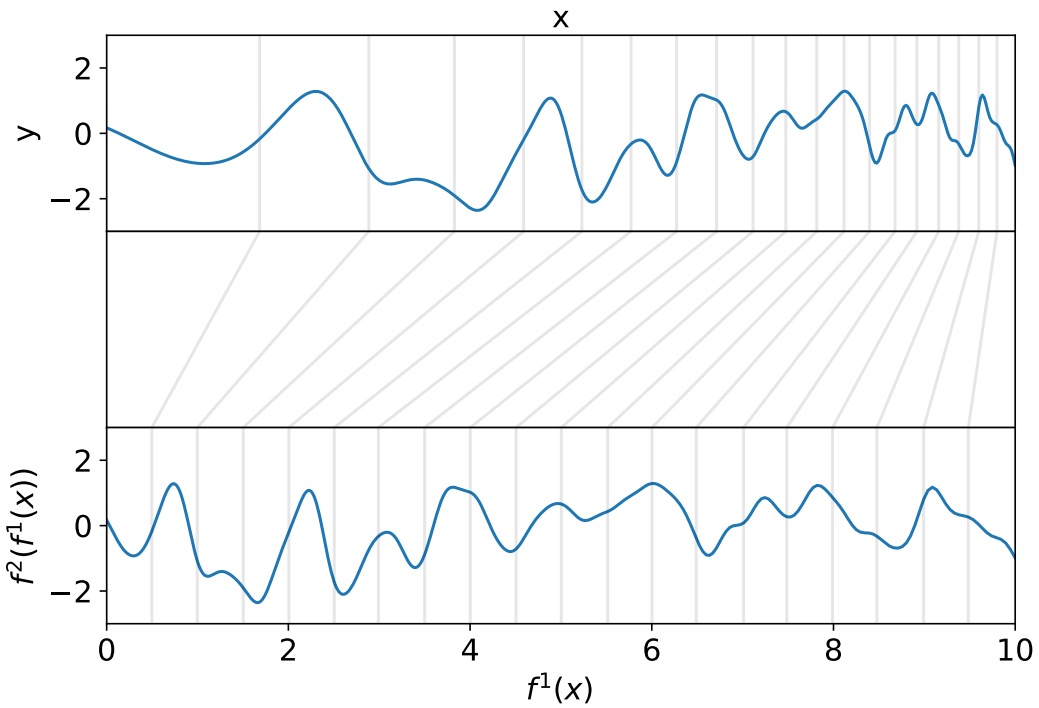


**Figure 1.2:** We show the idea behind a two-layer deep GP schematically: The top part of the figure shows the data $(x, y)$ that is to be modelled, which clearly has an x-dependent length scale. The first GP $f^1$ warps the inputs $x$ which is shown in the middle part of the figure. On the bottom we see that the second GP $f^2$ can then be used for regression on the warped inputs, where the data $(f^1(x), y)$ has a constant length scale.

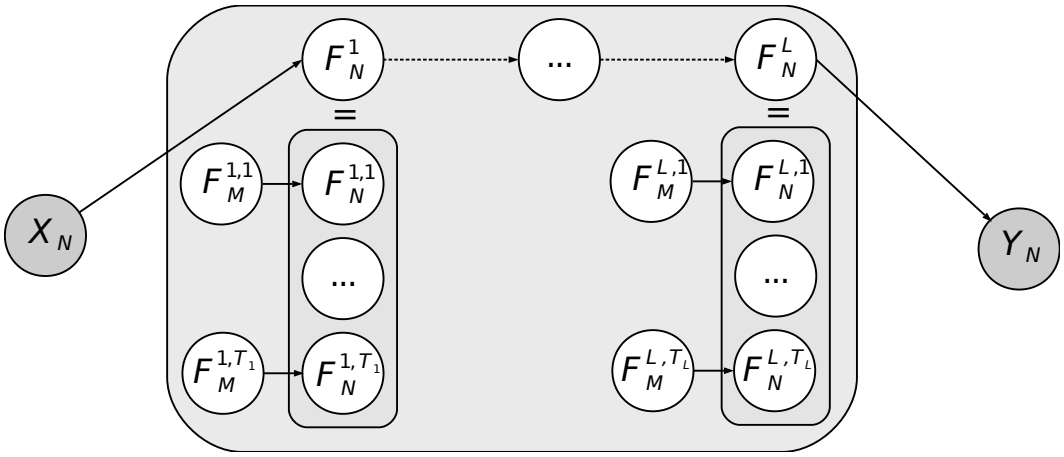


**Figure 1.3:** Plate diagram for an arbitrary deep GP. The inputs $X_N$ are mapped through $L$ layers. The $l$-th layer consists of $T_l$ sparse GPs with outputs $F_N^{l,t}$ that are conditioned on $F_M^{l,t}$ for $t = 1, \ldots, T_l$, respectively. From the outputs of the final layer $F_N^L$, we regress on the observed outputs $Y_N$.

leads to them being analytically marginalisable. The GP outputs $F_N^{l,t}$ remain analytically intractable and have to be recursively sampled after every layer in order to be propagated through the deep GP. This approach therefore uses *doubly stochastic variational inference* (Titsias and Lázaro-Gredilla, 2014) where the first source of stochasticity comes from recursively sampling the GP outputs and the second from subsampling the training data which is enabled through the use of sparse GPs.

The approach by Havasi et al. (2018) treats the GP outputs $F_N^{l,t}$ in the same way but uses a different approach in order to deal with the inducing outputs $F_M$. It employs stochastic gradient Hamiltonian Monte Carlo (Chen et al., 2014), a Markov chain Monte Carlo approach (see e.g. Neal, 1993), that is able to produce accurate samples from the exact posterior $p(F_M|Y_N)$ when using mini-batches. Hence, in contrast to the approach by Salimbeni and Deisenroth (2017) the work by Havasi et al. (2018) drops the independence and Gaussianity restrictions over the inducing outputs $F_M$ and in turn accepts the loss of their analytical tractability (having only access to samples).

Both state-of-the-art approaches have their advantages: Being able to analytically marginalise latent variables, as Salimbeni and Deisenroth (2017) do with the $F_M$, is known to be needed for fast convergence (Kingma et al., 2015). Taking correlations between latent variables into account when performing variational inference, as Havasi et al. (2018) do with the $F_M^{l,t}$, is important for calibrated uncertainty predictions (Turner and Sahani, 2011). Formulated differently, we could also state that each of the methods lacks the advantageous property of the respective other method. It is this observation that leads us to investigating inference methods for deep GPs further, trying to find compromises between the optimal convergence properties of Salimbeni and Deisenroth (2017) and the maximally expressive approach of Havasi et al. (2018). Is there some way to move along the Pareto-front defined by these two methods, i.e., can we sacrifice some of the expressivity of Havasi et al. (2018) to get improved converge properties? Or equivalently, can we add more expressivity to the approach of Salimbeni and Deisenroth (2017) while only sacrificing some of the good convergence properties? Together with the computational complexity of the approaches, these questions will be the focus of Chap. 3.

## 1.4 Introduction to Gaussian Process State-Space Models

As we discussed above in Sec. 1.2.3, GPSSMs aim at performing time series modelling for a time series $Y_T$ by combining state-space models with GPs. This amounts to introducing temporal latent states $X_T$ that disentangle the dynamics from the observation noise through the emission model $p(y_t|x_t)$ and that evolve according to a Markovian transition model $p(x_t|x_{t-1}, f_{t-1})$ (see Fig. 1.4).

Similarly as the deep GP above, this model is also not analytically tractable since the temporal latent states $X_T$ are inputs to the GP (hence to the non-linear kernel) and therefore cannot be marginalised out. Additionally, the task of inferring the GP modelling the transition function $f$ is very hard since by changing $f$, both the inputs ($x_{t-1}$) and the regression targets ($x_t$) for $f$ also change. Typically, this problem is approached by employing sparse GPs (introducing $\{X_M, F_M\}$), thus also enabling the application of GPSSMs to long time series. We schematically depict the resulting model in Fig. 1.4. The current state-of-the-art methods (Doerr et al., 2018; Ialongo et al., 2019) both use variational inference in order to perform approximate inference over the latent variables $X_T$, $F_M$, and $F_T \equiv \{f(x_t)\}_{t=1}^T$.

The approach by Doerr et al. (2018) uses a multivariate Gaussian distribution as the approximate posterior distribution $q(F_M)$ and assumes otherwise that the posterior distributions are given by the prior ones, e.g. $q(x_t|\cdot) = p(x_t|x_{t-1}, f_{t-1})$. Additionally, Doerr et al. (2018) use two other simplifying assumptions (which we discuss in detail in Sec. 2.4), leading to the analytical tractability of the $F_M$ and $F_T$, while the $X_T$ still cannot be marginalised analytically. Similarly to the treatment of deep GPs by Salimbeni and Deisenroth (2017) explained in Sec. 1.3, Doerr et al. (2018) therefore rely on doubly stochastic variational inference (Titsias and Lázaro-Gredilla, 2014): The first source of

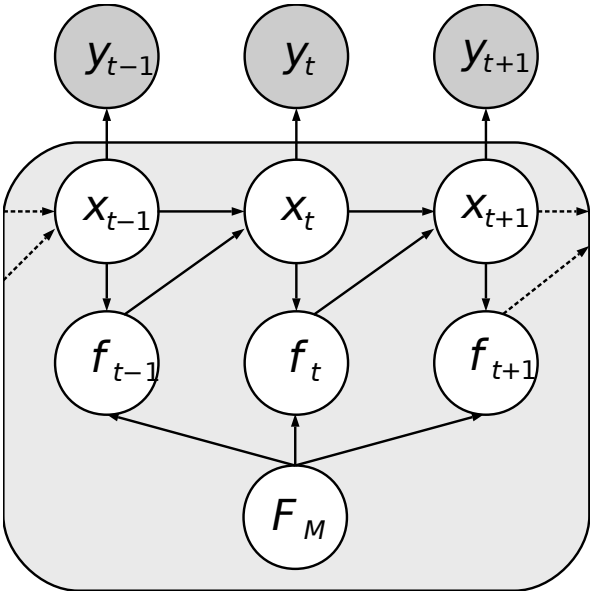


**Figure 1.4:** Plate diagram for the GPSSM: The observations $y_t$ depend only on the corresponding temporal latent state $x_t$. Those are influenced by the previous latent state $x_{t-1}$ and the GP output at the latter state, $f_{t-1} = f(x_{t-1})$. When using a sparse GP formulation, all GP outputs are additionally conditioned on the inducing outputs $F_M$.

stochasticity comes from sampling the $x_t$ after every time step, the second comes from being able to sample subsequences of the observations $Y_T$ through the use of the sparse GP.

The second approach by Ialongo et al. (2019), criticises the simplifying assumptions made by Doerr et al. (2018) and tries to find an inference method that works without those. This first step signifies theoretically more accurate inference but at the same time the loss of the analytical tractability of $F_M$ and $F_T$ both of which have to be additionally sampled instead. In a second step, Ialongo et al. (2019) employ a much more expressive posterior approximation over the $X_T$, assuming a parametric distribution $q(x_t|\cdot)$ that introduces a set of learnable parameters for every latent state $x_t$. The empirical evaluation of this more expressive method (which contains the method of Doerr et al. (2018) as a special case) is not fully convincing: In several experiments the more expressive method by Ialongo et al. (2019) is outperformed by the much simpler of Doerr et al. (2018). Once again, similarly as observed above for deep GPs in Sec. 1.3, we see a trade-off between expressivity and convergence properties, i.e., the ease of finding a good local optimum for the model parameters during training. In Chap. 6 of this thesis we will use this observation to design a new inference method for GPSSMs that moves along the Pareto-front defined by the methods of Doerr et al. (2018) and Ialongo et al. (2019), balancing their expressivity but working with as little parameters as possible to improve the convergence properties.

Additionally, there remains the question of the simplifying assumptions that Doerr et al. (2018) use and which Ialongo et al. (2019) criticise: Is this criticism fully valid and if so, can we still achieve similar theoretical results, i.e., the analytical tractability of the $F_M$ and $F_T$, to Doerr et al. (2018) without their simplifications? These questions along with their general relevance for probabilistic machine learning (and especially GPSSMs) will be the focus of Chap. 4 of this thesis.

Our findings there will provide the basis for the solution to a different problem that plagues both approaches to a different extent: the general applicability to long time series. The approach of Ialongo et al. (2019) is not designed to deal with mini-batches (or rather subsequences of the time series) and would therefore encounter problems with large runtimes and memory footprints as well as the problem of vanishing and exploding gradients (Pascanu et al., 2013) when being applied to long

trajectories. While the method of Doerr et al. (2018) can use subsequences, a general problem of using mini-batches for time series modelling persists: Slow effects that manifest only over a time spanned by multiple subsequence lengths can no longer be inferred (Williams and Zipser, 1995).[8] Using the theoretical results obtained from understanding the subtle differences between the two state-of-the-art GPSSM approaches in Chap. 4, we will design a new inference method that is able to overcome the aforementioned problems in Chap. 5.

## 1.5 Variational Inference for Composite Gaussian Process Models

As we have seen in the previous two sections, while seemingly very different models used for very different purposes (deep GPs can be used for flexible and expressive regression while GPSSMs are a specialised approach for time series modelling), there is a surprising number of similarities between deep GPs and GPSSMs. The most obvious similarity is that both are composite models where GPs play an important role: deep GPs are comprised of a small network of GPs while GPSSMs incorporate a GP in a state-space model. As we saw in the previous sections, all state-of-the-art approaches for deep GPs (Salimbeni and Deisenroth, 2017; Havasi et al., 2018) and GPSSMs (Doerr et al., 2018; Ialongo et al., 2019) build on sparse GPs to incorporate large data sets. Most of them therefore naturally also employ variational inference (all except for Havasi et al., 2018). Interestingly, the connection between the inference methods of Salimbeni and Deisenroth (2017) for deep GPs and of Doerr et al. (2018) for GPSSMs runs even deeper as both methods rely on doubly stochastic variational inference (Titsias and Lázaro-Gredilla, 2014). In this thesis we aim at exploiting this connection in order to further the theoretical understanding of inference methods in both models.

Furthermore, we saw that state-of-the-art approaches for both model classes trade off expressivity against efficiency (here loosely referring to either convergence properties or actual computational complexity). The central goal of this thesis is to explore these trade-offs further (guided by the gained theoretical understanding) by balancing efficiency against expressivity for inference in deep GPs and GPSSMs, thus designing new variational inference methods for composite Gaussian process models.

## 1.6 Thesis Outline and Publications

Above, we gave a short introduction to GPs and their sparse variants and introduced the general idea and the state-of-the-art methods for deep GPs and GPSSMs. We present further technical details and other related works for all these methods in Chap. 2. Then, in Chap. 3, we introduce a new inference method for deep GPs that trades off expressivity and computational complexity of state-of-the-art approaches. This inference method requires a deeper theoretical understanding of variational inference in deep GPs which we also provide.

Transfering this theoretical contribution to GPSSMs and advancing the understanding of subtle differences between the state-of-the-art approaches in this model class is the topic of Chap. 4. We then use this knowledge and, by exploiting connections to a related time series modeling approach, propose a new inference method that improves the efficiency of GPSSM approaches, especially for long time series data consisting of a composition of slowly and quickly changing signals (see Chap. 5). In Chap. 6 we present a different new inference method for GPSSMs that balances expressivity and convergence properties of state-of-the-art approaches by combining varational inference with another inference method that is often used for classical state-space models. Finally, in Chap. 7, we conclude and present an outlook.

[8] Existing methods that solve these problems for recurrent neural networks (Hochreiter and Schmidhuber, 1996; Chung et al., 2014) are not directly applicable to GPSSM models.

### 1.6.1 Publications and Contributions

This thesis contains material of multiple published conference articles that appear in a modified or extended form. These publications are listed below, along with information about the contributions of the authors and about where which parts of the publications appear in this thesis.

- Lindinger, J., Reeb, D., Lippert, C., & Rakitsch, B. (2020). Beyond the mean-field: Structured deep Gaussian processes improve the predictive uncertainties. *Advances in Neural Information Processing Systems*.

  The initial problem setting and idea were given by Barbara Rakitsch and David Reeb, while the details and experiments were designed by all authors jointly. Jakob Lindinger was responsible for working out the theoretical and technical details and implementing the algorithm. Jakob Lindinger and Barbara Rakitsch performed the experiments and analysed the results. Barbara Rakitsch and Jakob Lindinger wrote the main paper with contributions of all authors.

  Chap. 3 of the thesis closely follows this work.

- Longi, K., Lindinger, J., Duennbier, O., Kandemir, M., Klami, A., & Rakitsch, B. (2022). Traversing time with multi-resolution Gaussian process state-space models. *Annual Learning for Dynamics and Control Conference*
  and the accompanying technical report
  Longi, K., Lindinger, J., Duennbier, O., Kandemir, M., Klami, A., & Rakitsch, B. (2021). Traversing time with multi-resolution Gaussian process state-space models. *arXiv preprint arXiv:2112.03230*.

  The initial problem setting and idea were given by Barbara Rakitsch, while the details and experiments were designed by all authors jointly. Krista Longi was responsible for implementing the algorithm as well as performing the experiments and analysing the results. Jakob Lindinger derived the theoretical analysis of the method and was responsible for its write-up. Olaf Dünnbier provided the data set and domain knowledge for engine modeling. Barbara Rakitsch wrote the main paper with contributions of all authors.

  The main theoretical contribution of this publication can be found in Chap. 4 of the thesis, while some minor theoretical contributions along with the new inference method form the main part of Chap. 5.

  N.b. that I neither designed the inference method itself nor performed the experiments. These results are still included (although in a strongly altered description compared to the paper) in Secs. 5.2.1 and 5.3, respectively, as they provide the necessary motivation for my theoretical contributions.

- Lindinger, J., Rakitsch, B., & Lippert, C. (2022). Laplace approximated Gaussian process state-space models. *Conference on Uncertainty in Artificial Intelligence*.

  The initial problem setting and idea were given by Barbara Rakitsch, while the details and experiments were designed by all authors jointly. Jakob Lindinger was responsible for working out the theoretical and technical details and implementing the algorithm. Jakob Lindinger and Barbara Rakitsch performed the experiments and analysed the results. Jakob Lindinger and Barbara Rakitsch wrote the main paper with contributions of all authors.

  A small theoretical contribution from this paper can be found in Chap. 4 of the thesis, the rest builds the basis of Chap. 6.

CHAPTER 2

# Background

In this chapter we provide most of the necessary technical background for the remaining chapters of the thesis. We start in Sec. 2.1 by introducing the technical details about GP regression that we left out in Sec. 1.1. Then, in Sec. 2.2 we extend on the very short discussion about sparse GPs and variational inference in Sec. 1.2.1. In the final two sections of this chapter (Secs. 2.3 and 2.4), we continue the introduction to deep GPs and GPSSMs from Secs. 1.3 and 1.4, respectively, focussing more on a historical and technical perspective.

## 2.1 Gaussian Processes

In Sec. 1.1 we briefly introduced GPs as a probabilistic method for regression, i.e., the task of predicting outputs $y_*$ at unseen test inputs $x_*$ given a training set of $N$ input and output pairs $\{X_N, Y_N\}$. The underlying assumption for GP regression is that the mapping from inputs to outputs can be described by a function $f$ according to $y_n = f(x_n) + \epsilon_n$. Here, the observation noise is e.g. assumed to be independent and identically distributed (iid.) according to a zero-mean Gaussian, i.e., $\epsilon_n \sim \mathcal{N}\left(\epsilon_n \middle| 0, \sigma_y^2\right)$, with $\sigma_y^2$ being the observation noise variance. Taken together, this leads to the *likelihood*,

$$p(Y_N|F_N) = \prod_{n=1}^{N} \mathcal{N}\left(y_n \middle| f_n, \sigma_y^2\right) = \mathcal{N}\left(Y_N \middle| F_N, \sigma_y^2 I_N\right), \tag{2.1}$$

where $f_n \equiv f(x_n)$ and $I_N$ is the identity matrix of dimensionality $N$. Additionally, we place a zero-mean GP prior on the function $f$, resulting in the *prior* assumption $p(F_N) = \mathcal{N}(F_N|0, K_{NN})$ [Eq. $(1.1)$], where $F_N$ are the GP observations at the inputs $X_N$ and $K_{NN}$ is the covariance matrix obtained from evaluating the kernel $k$ at every pair of input points (see Sec. 1.1). The *joint density* of the model is then simply given by $p(Y_N, F_N) = p(Y_N|F_N)p(F_N)$.

In Bayesian inference (see e.g. Bishop, 2006, Sec. 1.2), the goal is to infer the *posterior* distribution $p(F_N|Y_N)$, i.e., the updated belief about the latent (unobserved) function values after having seen the data. The posterior can then be used to predict $y_*$ at unseen test points. Using Eqs. $(1.1)$ and $(2.1)$ and Gaussian calculus (see e.g. Toussaint, 2011), the posterior can be shown to equal (cf. Rasmussen and Williams, 2006, Sec. 2.3),[1]

$$p(F_N|Y_N) = \mathcal{N}\left(F_N \middle| K_{NN}\left(K_{NN} + \sigma_y^2 I_N\right)^{-1} Y_N, K_{NN} - K_{NN}\left(K_{NN} + \sigma_y^2 I_N\right)^{-1} K_{NN}\right). \tag{2.2}$$

We provide a detailed derivation of this equation (exemplarily for other similar derivations in this section) in Appx. A.1. In order to obtain a prediction from this posterior, we can exploit that under

[1] Note that here (and already in Eq. (1.1)) and in the following we generally omit to explicitly list the $X_N$ (which appear as inputs to the $K_{NN}$) as arguments of the distributions in order to have a less cluttered notation. Typically, the $X_N$ are merely constant inputs. Only in cases where important dependencies on $X_N$ require highlighting or when the $X_N$ are random variables, do they appear as arguments of the distributions.

a GP prior the function values at every finite set of input points are distributed according to a joint multivariate Gaussian distribution. In particular, this holds for $X_N \cup \{x_*\}$ which leads to

$$p(f_*|F_N) = \mathcal{N}\left(f_* \middle| K_{*N}K_{NN}^{-1}F_N, k_{**} - K_{*N}K_{NN}^{-1}K_{*N}^\top\right),$$

where $f_* \equiv f(x_*)$. Together with Eq. $(2.2)$ and the fact that $p(f_*|Y_N) = \int p(f_*|F_N)p(F_N|Y_N)dF_N$, this results in

$$p(f_*|Y_N) = \mathcal{N}\left(f_* \middle| K_{*N}\left(K_{NN} + \sigma_y^2 I_N\right)^{-1} Y_N, k_{**} - K_{*N}\left(K_{NN} + \sigma_y^2 I_N\right)^{-1} K_{*N}^\top\right), \quad (2.3)$$

where we used Gaussian calculus (see also Appx. A.1 for a more detailed derivation). The terms $k_{**} \equiv k(x_*, x_*)$ and $K_{*N} \equiv \{k(x_*, x_n)\}_{n=1}^N$ are evaluations of the kernel at different pairs of input points. Together with the fact that $p(y_*|Y_N) = \int p(y_*|f_*)p(f_*|Y_N)df_*$ and that $p(y_*|f_*) = \mathcal{N}\left(y_* \middle| f_*, \sigma_y^2\right)$ [cf. Eq. $(2.1)$], predicting for an observation $y_*$ instead of for a latent function value $f_*$ simply adds $\sigma_y^2$ to the variance of the right side of Eq. $(2.3)$:

$$p(y_*|Y_N) = \mathcal{N}\left(y_* \middle| K_{*N}\left(K_{NN} + \sigma_y^2 I_N\right)^{-1} Y_N, k_{**} - K_{*N}\left(K_{NN} + \sigma_y^2 I_N\right)^{-1} K_{*N}^\top + \sigma_y^2\right). \quad (2.4)$$

This formula can then be used to calculate predictions for arbitrary input points and has been used e.g. in Fig. 1.1 (bottom).

So far, we worked with the simplifying assumption that we are somehow given a kernel $k$ that works well with the data. But as we saw in Fig. 1.1 (top), the choice of the kernel has a large influence on how typical functions drawn from the GP with the respective kernel behave. Additionally, most kernels have a small number of parameters, the so-called hyperparameters, that further change how typical functions look like. As an example, we take the so-called radial basis function (RBF) kernel $k_{\text{RBF}}(x, x') = \sigma_k^2 e^{(x-x')^2/(2l^2)}$ that has two hyperparameters, the kernel variance $\sigma_k^2$ and the length scale $l$. Together with the noise variance $\sigma_y^2$, there are three hyperparameters $\theta = \{\sigma_k, l, \sigma_y\}$ that can be adjusted in a GP regression model with the RBF kernel. In Fig. 2.1, we demonstrate their influence by showing the GP prediction in Eq. $(2.4)$ three times for the same data set but fixing the length scale to three different values while adjusting the other two hyperparameters. While the GP with the correct length scale $l = 1$ (the groundtruth that has been used to generate the data) explains the data with a trade-off of signal and noise (top), the GP with a very short length scale (bottom left) is extremely flexible and explains all of the data by signal (leading to over-fitting) while the GP with $l = 3$ (bottom right) is forced to explain most of the data by noise (leading to under-fitting).

The setting of these kernel hyperparameters therefore also has a large influence on how well a GP is able to describe certain data sets. The typical way of setting the hyperparameters is to maximise the likelihood of the data under our model (see Rasmussen and Williams, 2006, Sec. 5.2), i.e., $p(Y_N) = \int p(Y_N, F_N)dF_N$, the *marginal*[2] (sometimes also called model evidence). For GP regression with a Gaussian likelihood the marginal is analytically tractable and can be computed using Eqs. $(1.1)$ and $(2.1)$ which yields $p(Y_N) = \mathcal{N}\left(Y_N \middle| 0, K_{NN} + \sigma_y^2 I_N\right)$. In order to find the optimal hyperparameters

[2] Called that way since the latent function values are marginalised/integrated out.

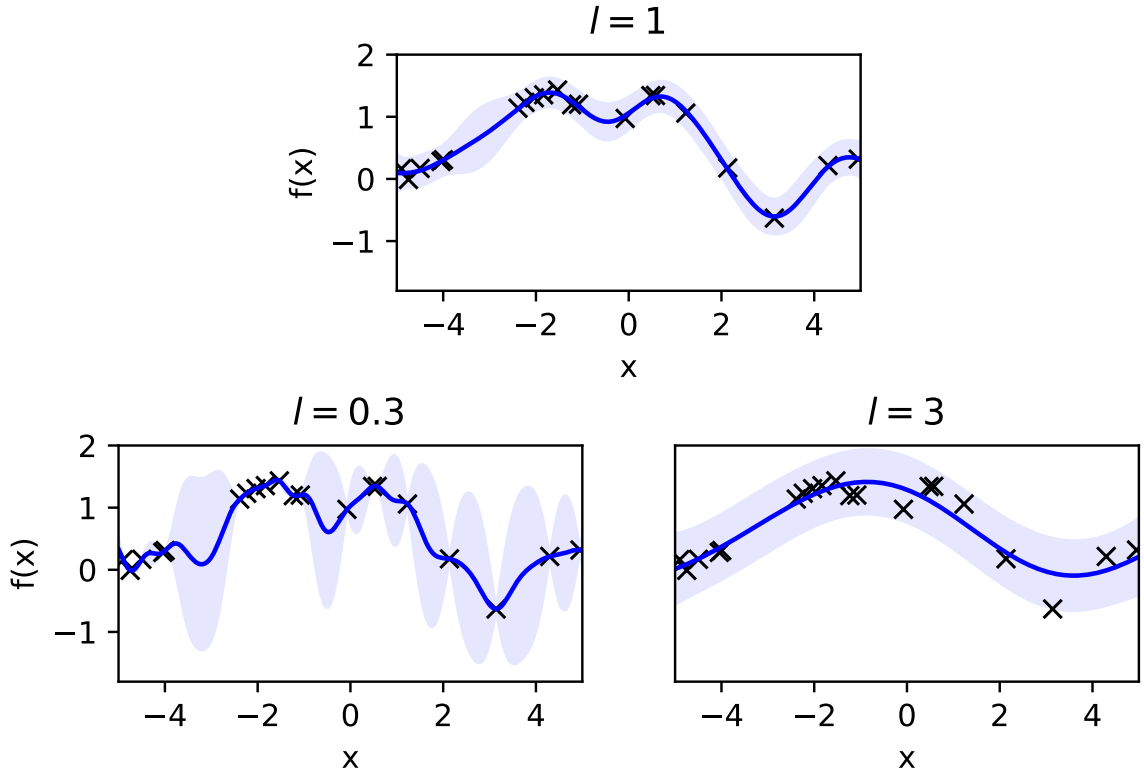


**Figure 2.1:** We depict the influence of the GP hyperparameters on the GP predictions. All plots show the same data set (black crosses) and the mean (solid blue line) and the confidence interval (two standard deviations, blue shaded areas) of different GP regression models. The top plot shows a model with the groundtruth hyperparameters which have been used to generate the data. The bottom plots show models where the length scale $l$ has been fixed to a wrong value (left: $l = 0.3$, right: $l = 3$) and the other hyperparameters have been optimised to provide the best fit under the respective constraint.

$\theta^*$, this formula, or rather its logarithm,[3]

$$\log p(Y_N) = -\frac{1}{2} Y_N^\top \left(K_{NN} + \sigma_y^2 I_N\right)^{-1} Y_N - \frac{1}{2} \log \det \left(K_{NN} + \sigma_y^2 I_N\right) + C, \tag{2.5}$$

can be optimised with respect to (wrt.) $\theta$. Note that in Eq. $(2.5)$ $C$ is a constant independent of $\theta$ and that the kernel hyperparameters appear only implicitly through $K_{NN}$. Furthermore, the terms on the right hand side (RHS) of Eq. $(2.5)$ have an intuitive interpretation: The first term is the data fit term (becoming larger the better the GP is able to describe the data) while the second is a complexity penalty term (becoming more negative the more flexible the kernel gets). Since generally Eq. $(2.5)$ cannot be optimised analytically wrt. $\theta$, numerical (gradient-based) methods have to be employed to find $\theta^*$. However, since the formula in Eq. (2.5) is typically not convex wrt. $\theta$, the obtained solution $\theta^*$ is usually only a local optimum and not the global one (see e.g. Nocedal and Wright, 1999).

Note that especially in small data regimes such as for Bayesian optimisation, a fully Bayesian treatment of the hyperparameters can be beneficial as it protects further from overfitting (see e.g. Snoek et al., 2012). This involves placing a hyperprior $p(\theta)$ on the hyperparameters and finding the corresponding posterior $p(\theta|Y_N)$. Generally, the latter cannot be calculated analytically such that ex-

[3] The logarithm is a concave transformation and therefore does not change the position at which the function is maximised. Using the logarithm helps preventing numerical problems coming from values extremely close to zero that the marginal likelihood can take on.

pensive Markov chain Monte Carlo approaches (e.g. Murray and Adams, 2010) or other approximate inference methods have to be used.

## 2.2 Sparse Gaussian Processes and Variational Inference

The GP regression approach that we introduced in the previous section, while being a powerful probabilistic method, does not scale to modern, large datasets. Inverting the $N \times N$ matrix $K_{NN}$ required for predictions [Eq. $(2.4)$] and calculating the log-determinant of $K_{NN}$ which is additionally required for model selection [Eq. $(2.5)$] leads to a cubic scaling in the number of data points $N$. This practically limits the application of GP regression to data sets with at most a few thousand data points. Shortly after the introduction of GPs to the machine learning community, this problem was realised and has since been the focus of a multitude of works on GPs. In this section we first provide an overview over the history leading up to sparse variational GPs (Sec. 2.2.1) before thoroughly introducing this method in more detail in Sec. 2.2.2 as this latter approach is then used throughout this thesis. At the end of this section we give a small (non-exhaustive) overview over other approaches to sparse or scalable GPs (Sec. 2.2.3).

### 2.2.1 Inducing Point Approximations

Early approaches to sparse GPs included very simple solutions like taking a subset of $M$ data points out of the full data set with $N$ data points to perform GP regression (see e.g. Hayashi et al., 2020, for a recent theoretical analysis). This reduces the computational complexity to $\mathcal{O}(M^3)$ but naturally fails to include information from the whole data set. More sophisticated approaches built upon the Nyström approximation, $K_{NN} \approx K_{NM}K_{MM}^{-1}K_{NM}^\top$, which also uses a subset of the data points of size $M$, $X_M = \{x_m\}_{m=1}^M$ and where $K_{MM} = \{k(x_m, x'_m)\}_{m,m'=1}^M$. However, these approaches take the relation to the rest of the data points through the kernel matrix $K_{NM} = \{k(x_n, x_m)\}_{n,m=1}^{N,M}$ into account. For $M \ll N$ this reduces the computational complexity to $\mathcal{O}(NM^2)$. However, naively using this approximation in a GP (Williams and Seeger, 2000) does not lead to a valid probabilistic method as the variances of such an approach can become negative (Williams et al., 2002).

Different approaches based on so-called *inducing points* (e.g. Quinonero-Candela and Rasmussen, 2005) lead to a similar formulation as the Nyström approximation, but are motivated from a different perspective: They start by introducing a set of inducing points $\{X_M, F_M\}$, where the *inducing inputs* $X_M$ are not necessarily a subset of $X_N$ and the *inducing outputs* $F_M \equiv \{f(x_m)\}_{m=1}^M$ are the values of the GP evaluated at the inducing inputs. Due to the property that the GP outputs for every finite set of inputs share a joint multivariate Gaussian distribution [cf. Eq. $(1.1)$], the same holds true for the input set $X_N \cup X_M$. The joint distribution over the corresponding outputs can be written as $p(F_N, F_M) = p(F_N|F_M)p(F_M)$ with (see e.g. Toussaint, 2011),

$$p(F_M) = \mathcal{N}(F_M|0, K_{MM}), \tag{2.6}$$

$$p(F_N|F_M) = \mathcal{N}(F_N|\mu(X_N, F_M), \Sigma(X_N)), \tag{2.7}$$

$$\mu(X_N, F_M) = K_{NM}K_{MM}^{-1}F_M, \qquad \Sigma(X_N) = K_{NN} - K_{NM}K_{MM}^{-1}K_{NM}^\top, \tag{2.8}$$

where the Nyström approximation term appears in the covariance $\Sigma$ in Eq. $(2.8)$. Note that so far this is still an exact expression: We could analytically marginalise out the pseudo-data from the augmented prior [Eqs. $(2.6)$ and $(2.7)$], resulting again in the standard prior [Eq. $(1.1)$], i.e., $\int p(F_N|F_M)p(F_M)dF_M = p(F_N)$. However, for making predictions $f_*$ at unseen test points $x_*$ this inducing point sparse GP approach assumes that $F_M$ is a sufficient statistic for $F_N$ which implies

the approximation $p(f_*|F_N, F_M) \approx p(f_*|F_M)$.[4] This assumption therefore leads to the prediction formula[5]

$$p(f_*|Y_N) \approx \int p(f_*|F_M)p(F_M|Y_N)dF_M, \tag{2.9}$$

where $p(F_M|Y_N)$ is the posterior over the inducing outputs [cf. Eq. (2.2)].

Early approaches to sparse GPs using inducing points (Smola and Bartlett, 2000; Seeger et al., 2003; Snelson and Ghahramani, 2005) were later interpreted under a unifying framework and shown to propose different approximations to the prior conditional in Eq. (2.7) (Quinonero-Candela and Rasmussen, 2005). One prominent method that falls in this category and that we build on in the course of this thesis is called fully independent training conditional (FITC, Snelson and Ghahramani, 2005). Quinonero-Candela and Rasmussen (2005) showed that the latter method is equivalent to the assumption that the GP outputs in Eq. (2.7) are fully independent given the inducing outputs, i.e.,

$$p(F_N|F_M) \approx \prod_{n=1}^{N} p(f_n|F_M) = \mathcal{N}\left(F_N \middle| K_{NM}K_{MM}^{-1}F_M, \operatorname{diag}\left[K_{NN} - K_{NM}K_{MM}^{-1}K_{NM}^{\top}\right]\right), \tag{2.10}$$

where $\operatorname{diag}[A]$ denotes the operation that returns a diagonal matrix containing only the diagonal elements of the matrix $A$. In Fig. 2.2 (top) we compare the predictions made by this method [which can be obtained by following the same derivation as in Sec. 2.1 but using Eq. (2.9)] with those of a standard GP.

### 2.2.2 Sparse Variational Gaussian Processes

Instead of approximating the prior, later works have used the full augmented prior in Eqs. (2.6) and (2.7) and attempted to approximate the posterior $p(F_N, F_M|Y_N)$. The most prominent approaches (Titsias, 2009; Hensman et al., 2013) are based on *variational inference* (see e.g. Blei et al., 2017) and still in use today. We thoroughly introduce these approaches after giving a short introduction to variational inference in the following.

Variational inference aims at finding an approximation $q(F_N, F_M)$ to $p(F_N, F_M|Y_N)$, the true posterior. This is done by first choosing a parametric variational family for the distribution $q_\psi(F_N, F_M)$ with parameters $\psi$. Next, we find the optimal setting of the parameters such that the approximate posterior $q_\psi(F_N, F_M)$ is as close as possible to $p(F_N, F_M|Y_N)$, where closeness is measured by the (reverse) Kullback-Leibler (KL) divergence. It can be shown that maximising the so-called evidence lower bound (ELBO),

$$\mathcal{L}(\psi) \equiv \int q_\psi(F_N, F_M) \log \frac{p(Y_N, F_N, F_M)}{q_\psi(F_N, F_M)} dF_N dF_M, \tag{2.11}$$

is equivalent to minimising the KL divergence. We provide a more detailed introduction, including a derivation of Eq. (2.11) in Appx. A.2.

Returning to the problem of sparse GPs,[6] we have $p(Y_N, F_N, F_M) = p(Y_N|F_N)p(F_N|F_M)p(F_M)$ as the prior, where the terms are given in Eqs. (2.1), (2.6) and (2.7). The conventional variational

[4] The name *inducing* is derived from the fact that the pseudo-data $\{X_M, F_M\}$ are assumed to *induce* a distribution on the GP predictions $p(f_*|F_M)$.

[5] Derivation: $p(f_*|Y_N) = \int p(f_*|F_N, F_M)p(F_N, F_M|Y_N)dF_M dF_N \approx \int p(f_*|F_M)p(F_N, F_M|Y_N)dF_M dF_N = \int p(f_*|F_M)p(F_M|Y_N)dF_M$, where the first step is the definition, the next step uses the sufficient statistic assumption and the last step is a standard marginalisation property [Eq. (A.9)].

[6] There are a lot of subtle points to be considered when using variational inference for minimising KL divergences between stochastic processes (which GPs are). This is discussed in detail in Matthews et al. (2016).

family for this problem that Titsias (2009) introduced is given by

$$q(F_N, F_M) = p(F_N|F_M)q(F_M). \tag{2.12}$$

Choosing the prior conditional $p(F_N|F_M)$ as the conditional distribution is crucial as it leads to cancellation of that expensive term when plugging the prior and Eq. (2.12) into Eq. (2.11). In this particular problem it turns out that the variational distribution over the inducing outputs $q(F_M)$ can be analytically optimised resulting in a multivariate Gaussian distribution with known mean and covariance (see Titsias, 2009). The calculations necessary for this scale as $\mathcal{O}(NM^2)$ for $M \ll N$ and the memory requirements as $\mathcal{O}(NM)$. For very large data sets this is unfortunately still insufficient.

With a subtle change in the formulation of Titsias (2009), Hensman et al. (2013) achieved the applicability of sparse GPs also to very large data sets: Instead of calculating the optimal $q(F_M)$, we can parametrise this distribution as $q_\psi(F_M) = \mathcal{N}(F_M|\mu_M, S_M)$ with variational parameters $\psi = \{\mu_M, S_M\}$. Plugging this together with Eq. (2.12) and the prior into Eq. (2.11) leads after some manipulation (which we detail in Appx. A.2.1) to

$$\mathcal{L}_{\text{SGP}}(\psi) = \sum_{n=1}^{N} \mathbb{E}_{q(f_n)}\left[\log p(y_n|f_n)\right] - \text{KL}[q_\psi(F_M)||p(F_M)], \tag{2.13}$$

the ELBO for sparse GPs. Here, $\mathbb{E}$ denotes an expectation value that is defined as

$$\mathbb{E}_{p(x)}\left[g(x)\right] \equiv \int p(x)g(x)dx, \tag{2.14}$$

for an arbitrary distribution $p$ of a variable $x$ and a function $g$. Furthermore, the $q(f_n)$ are the marginals of the distribution $q(F_N, F_M)$ which are obtained by analytically marginalising out the inducing outputs $F_M$ and all $f_{n'}$ for $n' \neq n$, resulting in

$$q(f_n) = \iint q(F_N, F_M)dF_M \prod_{n' \neq n} df_{n'} = \mathcal{N}(f_n|\mu_n, \Sigma_n), \tag{2.15}$$

$$\mu_n = K_{nM}K_{MM}^{-1}\mu_M, \qquad \Sigma_n = k_{nn} - K_{nM}K_{MM}^{-1}\left(K_{MM} - S_M\right)K_{MM}^{-1}K_{nM}^{\top}. \tag{2.16}$$

Inference in this model is performed by optimising Eq. (2.13) wrt. $\psi$. Since the first term on its RHS can be written as a sum over the data points while the other term is independent of the data, we can optimise the ELBO stochastically, i.e., by taking mini-batches from $Y_N$ of size $N_b < N$. This reduces the computational complexity to $\mathcal{O}(N_b M^2 + M^3)$, which for typical choices with $M$ of $\mathcal{O}(10^2)$ and $N_b$ of $\mathcal{O}(10^3)$ is easily computationally tractable.[7] The change from the formulation of Titsias (2009) to the one of Hensman et al. (2013) therefore amounts to a trade-off: Instead of analytically finding the optimal distribution $q(F_M)$ for the whole data set in one step (which has a large computational complexity), we use subsets of the data and approximate the optimal distribution $q(F_M)$ slowly with many small steps (each of which has a low computational complexity).

In addition to the variational parameters $\psi$, we still have to optimise the hyperparameters $\theta$ of the model. For standard GPs in Sec. 2.1, we did this by optimising the log-evidence in Eq. (2.5) wrt. $\theta$. Conveniently, the ELBO in Eq. (2.13) is a lower bound to the log-evidence [cf. Eq. (A.11)], and in principle also depends on the setting of the hyperparameters which we denote as $\mathcal{L}_{\text{SGP}}(\psi, \theta)$. Hence by optimising $\mathcal{L}_{\text{SGP}}(\psi, \theta)$ wrt. both $\psi$ and $\theta$ we simultaneously perform approximate inference and

[7] Note that the choice of the number of inducing points $M$ and the initialisation of the inducing inputs $X_M$ can have a large impact on the performance of the trained model. In a recent work, Burt et al. (2019) give theoretically grounded advice on both these points.

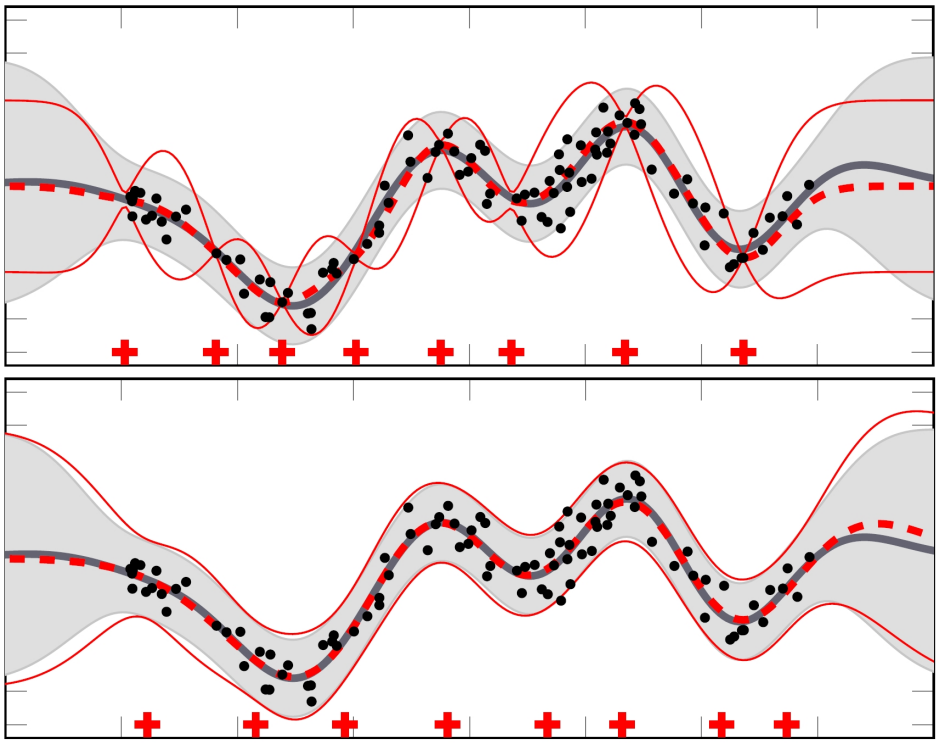

**Figure 2.2:** We show a comparison of predictions of sparse GP approaches to those of a standard GP. Both plots depict the same data (black circles) and the mean (grey solid line) and confidence interval (grey shaded area) of a standard GP regression model. While the top plot shows an approximate model using the FITC approximation (Snelson and Ghahramani, 2005), the bottom plot shows a variational sparse GP (Titsias, 2009). For both approaches, all model hyperparameters (including number of the inducing points and the location of the inducing inputs) have been optimised. The dashed (solid) red lines show the mean (confidence intervals) for the approximate GP predictions while the red crosses show the optimised inducing input locations. Taken from Bauer et al. (2016).[8]

approximate model selection.[9] Predictions in this model can then be performed by using Eq. (2.9) but replacing $p(F_M|Y_N)$ with the approximate posterior $q_\psi(F_M)$. This results in $q(f_*) = \mathcal{N}(f_*|\mu_*, \Sigma_*)$, where $\mu_*$ and $\Sigma_*$ are given by the equivalents of Eq. (2.16), i.e., where every occurrence of $n$ has to be replaced by $*$. In Fig. 2.2 (bottom) we show a comparison of predictions using this approach with a standard GP and also with the FITC approach (top) from the previous section. For a more in-depth comparison between the two approximate approaches we refer the interested reader to Bauer et al. (2016).

### 2.2.3 Other Approaches to Sparse Gaussian Process

Even though we only build on the approaches to sparse GPs that are introduced above, we still want to spend some effort in this section on introducing selected other sparse GPs variants. An extensive recent review is provided in Liu et al. (2020).

An approach that is equally intuitive as simply using a subset with $M$ data points of the whole data

[8] Note that the FITC approximation does not generally reduce the noise variance to almost zero (which is the case for Fig. 2.2 (top), where there is almost no variance at the position of the inducing inputs), it however shows a tendency for small noise variances. See also the discussion and other plots in Bauer et al. (2016).

[9] Note that for a Gaussian likelihood and a Gaussian variational distribution all terms in $\mathcal{L}_{\text{SGP}}(\psi, \theta)$ [Eq. (2.13)] are analytically tractable (see e.g. Hensman et al., 2013). Furthermore, see e.g. Adam et al. (2021) for existing best practices for the joint optimisation of $\theta$ and $\psi$ and a recent improvement on them.

set with $N$ data points, is to separate the input space into $N/M$ disjoint regions, for each of which a so-called local expert GP is responsible (see e.g. Kim et al., 2005; Datta et al., 2016). This leads to each of the GPs being assigned roughly $M$ data points such that training all GPs has complexity $\mathcal{O}(NM^2)$, similarly as the methods introduced above. Unfortunately, this approach leads to discontinuities on the boundaries of the expert regions for which two directions of research have proposed solutions: First, the so-called mixtures of experts (see e.g. Tresp, 2000b; Rasmussen and Ghahramani, 2001; Yuksel et al., 2012) which models the likelihood $p(Y_N|X_N)$ as a Gaussian mixture and second, the product of experts (see e.g. Tresp, 2000a; Hinton, 2002; Deisenroth and Ng, 2015) which uses a product of the individual experts' likelihood. The review by Liu et al. (2020) discusses the differences and advantages of these two model classes in great detail.

Another direction of work proposes placing the inducing points in variational sparse GPs on a grid which makes the approach amenable to scalable Kronecker product approximations (Wilson and Nickisch, 2015). In combination with tensor-train decompositions (Oseledets, 2011) and deep kernel learning (Wilson et al., 2016; Calandra et al., 2016) this leads to a method that can deal with billions of inducing points and data sets with a high input-dimensionality (Izmailov et al., 2018).

A further line of work deals with making exact GPs applicable to large data sets by utilising efficient and parallelisable methods (such as using conjugate-gradient instead of matrix inversions) to perform the calculations on multiple GPUs, culminating in the applicability of exact GPs to a data set with a million data points (see Wang et al., 2019, and references therein). A recent addition combines these ideas with low-precision arithmetic to increase the scalability even further (Maddox et al., 2022).

In Särkkä et al. (2013), the authors rephrase GP regression for certain (spatio)temporal problems as a state-space model that can be solved with Kalman filtering and smoothing (Särkkä, 2013) in linear (instead of cubic) time in the number of time points.

Finally, there is also work that generalises the notion of the inducing outputs, making it able to place the inducing inputs in an arbitrary space such that additional prior knowledge can be considered to sparsify GPs. This methodology is known as inter-domain inducing features (Lázaro-Gredilla and Figueiras-Vidal, 2009). A particularly scalable approach is achieved when using the Fourier transform to obtain the inducing outputs (Hensman et al., 2017). This method has recently been combined by Wilson et al. (2020) with the variational sparse GP of Hensman et al. (2013) to provide an approach that also allows efficient sampling from the approximate posterior which is for example important in Bayesian optimisation (Shahriari et al., 2016).

## 2.3 Technical Background on Deep Gaussian Processes

As we discussed in Sec. 1.2.2, GPs or their sparse variants that we introduced in detail in the previous two sections might need additional kernel or feature engineering to be optimally applied to certain data sets. Both of these methods require expert knowledge and are prone to overfitting, whereas deep GPs promise to be a very flexible regression approach without the need for kernel or feature engineering. While we discussed the idea behind deep GPs and the difference between state-of-the-art approaches in Sec. 1.3, here we provide also a short overview over other works dealing with inference in deep GPs, and the necessary technical background. As we will see, all deep GP inference methods build on the sparse GP frameworks introduced in the previous section.

Deep GPs have been introduced by Damianou and Lawrence (2013) as a model that maps input point $x_n$ to observations $y_n$ via a mapping with $L$ layers, $y_n = f^L(\cdots f^1(x_n)) + \epsilon_n$, where we place GP priors on all functions $f^l$ and the noise $\epsilon_n$ is typically chosen to be iid. Gaussian distributed. The functions $f^l$ are represented as layers of a small network of GPs (see Fig. 1.3) and the concatenation of the outputs of all GPs in one layer are treated as the inputs for the GPs in the next layer, e.g. a layer containing five GPs produces a five-dimensional output. Formally, the $l$-th layer output for an

input point $x_n$ can be written as $f_n^l = [f^{l,1}(f_n^{l-1}), \dots, f^{l,T_l}(f_n^{l-1})]^\top$ which is recursively defined for $l = 1, \dots, L$ starting with $f_n^1 = [f^{1,1}(x_n), \dots, f^{1,T_1}(x_n)]^\top$. Here we defined $f_n^0 \equiv x_n$, and $T_l$ is the number of GPs (tasks) in the $l$-th layer and $f^{l,t}$ is the $t$-th GP in the $l$-th layer of the deep GP. For a full data set $\{X_N, Y_N\}$, this model results in $TN$ latent GP observations (where $T \equiv \sum_{l=1}^L T_l$ is the total number of GPs in the deep GP) which we summarise in $F_N \equiv \{F_N^l\}_{l=1}^L$, where the $F_N^l = \{F_N^{l,t}\}_{t=1}^{T_l}$ contain the layer-wise observations and the $F_N^{l,t} \equiv \{f_n^{l,t}(f_n^{l-1})\}_{n=1}^N$ those for all data points for a single GP. Placing independent zero-mean Gaussian priors with kernels $k^{l,t}$ on all functions $f^{l,t}$ leads together with the recursive formulation to the factorisation $p(F_N) = \prod_{l=1}^L p(F_N^l|F_N^{l-1})$. Here we have $p(F_N^l|F_N^{l-1}) = \prod_{t=1}^{T_l} \mathcal{N}\left(F_N^{l,t}\middle|0, K_{NN}^{l,t}\right)$, where $K_{NN}^{l,t} = \{k^{l,t}(f_n^{l-1}, f_{n'}^{l-1})\}_{n,n'=1}^N$. According to our modelling assumption $y_n = f^L(\cdots f^1(x_n)) + \epsilon_n$, with $\epsilon_n$ iid. Gaussian distributed with noise variance $\sigma_y^2$, the likelihood of this model is given by [cf. Eq. (2.1)]

$$p(Y_N|F_N^L) = \mathcal{N}\left(Y_N\middle|F_N^L, \sigma_y^2 I_N\right). \tag{2.17}$$

Inference in the joint model $p(Y_N, F_N) = p(Y_N|F_N^L)\prod_{l=1}^L p(F_N^l|F_N^{l-1})$ is intractable since the outputs of one layer are inputs to the (non-linear) kernel of the next layer. In order to deal with this and also to be able to scale to large data sets, the sparse GP formalism from Sec. 2.2.2 is applied to this model: We augment every GP prior with a set of inducing points $\{X_M^{l,t}, F_M^{l,t}\}$ leading to the augmented prior model [cf. the plate diagram in Fig. 1.3]

$$\begin{aligned} p(Y_N, F_N, F_M) &= p(Y_N|F_N^L)\prod_{l=1}^L p(F_N^l|F_M^l, F_N^{l-1})p(F_M^l), \\ p(F_N^l|F_M^l, F_N^{l-1}) &= \prod_{t=1}^{T_l} p(F_N^{l,t}|F_M^{l,t}, F_N^{l-1}), \quad p(F_M^l) = \prod_{t=1}^{T_l}\mathcal{N}\left(F_M^{l,t}\middle|0, K_{MM}^{l,t}\right). \end{aligned} \tag{2.18}$$

Here $K_{MM}^{l,t} = \{k^{l,t}(x_m^{l,t}, x_{m'}^{l,t})\}_{m,m'=1}^M$ and $p(F_N^{l,t}|F_M^{l,t}, F_N^{l-1})$ is as in Eq. (2.7) with the $K$ matrices replaced by their $K^{l,t}$ counterparts, where e.g. $K_{NM}^{l,t} = \{k^{l,t}(f_n^{l-1}, x_m^{l,t})\}_{n,m=1}^{N,M}$. Additionally, the $F_M$, $F_M^l$ and $F_M^{l,t}$ contain the collection of all inducing outputs in the deep GP, those in layer $l$, and only those in the $t$-th GP in the $l$-th layer, respectively.

In order to find an analytically tractable approximate inference scheme, Damianou and Lawrence (2013) introduce additional latent variables $X_N^l$ per layer which are assumed to be noisy observations of the $F_N^l$. They then proceed to perform variational inference with a variational family that assumes independence between the $X_N^l$ and the $F_N^l$. While this leads to an analytically tractable solution, strong independence assumptions in variational inference are known to be problematic: Especially for strongly correlated variables (which we expect the $X_N^l$ and the $F_N^l$ to be since the first is assumed to be a noisy observation of the latter), assuming them to be independent can lead to a strong underestimation of the true marginal probabilities (Turner and Sahani, 2011). Furthermore, the variational posterior has lost all correlations between different layers due to this independence assumption, leading to some of the layers being practically turned off, i.e. the signal to noise ratio goes towards zero during model selection (see Salimbeni and Deisenroth, 2017 and also the discussion therein for further information). There have been several works that proposed twists or improvements to the inference method by Damianou and Lawrence (2013), but initially all approaches still built on the same independence assumption: In Hensman and Lawrence (2014) a nested variational compression approach is proposed, while Dai et al. (2016) use a variational auto-encoder. Furthermore, Bui et al. (2016) introduce an approximate expectation propagation approach that additionally builds on the FITC approximation for the sparse GP [Eq. (2.10)] and Cutajar et al. (2017) propose using random

feature expansions to improve the scalability of deep GPs.

The first approach that takes the dependence between successive layers in a variational approach into account is the one by Salimbeni and Deisenroth (2017): They first note that the same effect as introducing $X_N^l$ as a noisy version of the $F_N^l$ can similarly be achieved by instead introducing an additional white noise kernel component in the kernels $k^{l,t}$.[10] Variational inference in the deep GP model given in Eq. (2.18) can then be performed using the variational posterior

$$q(F_N, F_M) = q(F_M) \prod_{l=1}^{L} p(F_N^l | F_M^l, F_N^{l-1}). \tag{2.19}$$

Note that this is the same general form as the one used for sparse GPs [Eq. (2.12)]. With this choice, the ELBO has the form

$$\mathcal{L}_{\text{DGP}} = \sum_{n=1}^{N} \mathbb{E}_{q(f_n^L)} \left[ \log p(y_n | f_n^L) \right] - \text{KL}[q(F_M) || \prod_{l=1}^{L} p(F_M^l)], \tag{2.20}$$

which we derive in Appx. A.2.2. Note that this ELBO looks almost identical to the one of the variational sparse GPs in Eq. (2.13). The difference lies in the last layer marginals $q(f_n^L)$ which have a much more complicated form,

$$q(f_n^L) = \int \left[ \int q(F_M) \prod_{l=1}^{L} p(f_n^l | F_M^l, f_n^{l-1}) dF_M \right] df_n^1 \cdots df_n^{L-1}, \tag{2.21}$$

where the explicit formula for the marginals $p(f_n^l | F_M^l, f_n^{l-1})$ of the $p(F_N^l | F_M^l, F_N^{l-1})$ is provided in Eq. (A.21). Compared to the equivalent term for sparse GPs in Eq. (2.15), we not only have to marginalise out the inducing outputs and GP outputs for one GP, but rather for all GPs in all layers of the deep GP in Eq. (2.21).

In order to achieve analytical tractability of the inner integral in Eq. (2.21), Salimbeni and Deisenroth (2017) choose a mean-field (MF) Gaussian approximation for $q(F_M)$, the variational distribution over the inducing outputs:

$$q_{\text{MF}}(F_M) = \prod_{l=1}^{L} \prod_{t=1}^{T_l} q_\psi(F_M^{l,t}), \qquad q_\psi(F_M^{l,t}) = \mathcal{N}\left( F_M^{l,t} \middle| \mu_M^{l,t}, S_M^{l,t} \right). \tag{2.22}$$

The variational parameters $\psi$ are therefore given as the mean vectors $\mu_M^{l,t}$ and the covariance matrices $S_M^{l,t}$ of all the GPs in the deep GP. The form of the variational distribution in Eq. (2.22) allows for an analytical solution of the inner integral in Eq. (2.21) (see Appx. A.2.2), resulting in

$$q_{\text{MF}}(f_n^L) = \int \prod_{l=1}^{L} q(f_n^l | f_n^{l-1}) df_n^1 \cdots df_n^{L-1}, \tag{2.23}$$

where $q(f_n^l | f_n^{l-1}) = \prod_{t=1}^{T_l} \mathcal{N}\left( f_n^{l,t} \middle| \mu_n^{l,t}, \Sigma_n^{l,t} \right)$ and the means and covariances are given by the equivalents of Eq. (2.16). Note that the remaining integral in Eq. (2.23) is intractable. In Salimbeni and Deisenroth (2017), this is solved by recursively sampling the marginals through the layers of the deep GP, i.e., starting with $f_n^{1(s)} \sim q(f_n^1 | x_n)$ until $f_n^{L(s)} \sim q(f_n^L | f_n^{L-1(s)})$, where the index $(s)$ indicates

[10] In the experiments Salimbeni and Deisenroth (2017) note that this component has no effect on the performance and thus can be omitted.

a sampled quantity. By reparametrising these samples (Kingma and Welling, 2014; Rezende et al., 2014; Titsias and Lázaro-Gredilla, 2014) it is possible to calculate unbiased gradients wrt. the variational parameters $\psi$ for optimising the ELBO in Eq. $(2.20)$. To this first source of stochasticity comes the possibility of subsampling the data since the bound in Eq. $(2.20)$ is a sum over the independent data points, hence leading to the doubly stochastic variational inference approach by Salimbeni and Deisenroth (2017).

In order to make this approach work in practice, Salimbeni and Deisenroth (2017) introduce two small changes to the general approach described so far: First, the number of trainable parameters is reduced by using the same kernel $k^l$ and the same inducing inputs $X_M^l$ for all GPs within the same layer $l$, which simplifies finding a good local optimum for the inducing variables and the kernel hyperparameters. Second, the authors propose using linear mean functions for all inner GPs in the deep GP given by the principal component analysis (see e.g. Bishop, 2006, Sec. 12.1) mapping of the original inputs $X_N$. These mean functions are needed to avoid pathologies found in deep GPs (Duvenaud et al., 2014), while their form is inspired by skip connections in residual neural networks (He et al., 2016). Finally, predictions in this model can be made by taking multiple samples from $q(f_*^L)$ using Eq. $(2.23)$, where the training input is being replaced by the test input $x_*$. The resulting predictive distribution is a mixture of Gaussians that can in principle approximate arbitrarily complex distributions. In summary, the approach by Salimbeni and Deisenroth (2017) trades off losing the analytical tractability during inference versus removing independence assumptions in the approximate posterior compared to the work by Damianou and Lawrence (2013).

In a follow-up work, Havasi et al. (2018) sacrifice yet another part of analytical tractability in order to significantly improve the expressivity of the approach by Salimbeni and Deisenroth (2017): The authors criticise the Gaussian mean-field assumption made for $q(F_M)$ in Eq. $(2.22)$ both for the independence assumption between layers and the simplicity of such an approach that e.g. does not allow for multimodality. Instead, they propose to approximately marginalise the $F_N$ from the model in Eq. $(2.18)$ using the same recursive sampling scheme as Salimbeni and Deisenroth (2017) but with $p(f_n^l|F_M^l, f_n^{l-1})$ instead of $q(f_n^l|f_n^{l-1})$ leaving them with an approximation to $p(Y_N, F_M)$. Havasi et al. (2018) then propose to use a specialised Markov chain Monte Carlo approach (Neal, 1993; Chen et al., 2014) in order to produce samples from the posterior $p(F_M|Y_N)$ which can then be used to make predictions. Hence, in comparison to Salimbeni and Deisenroth (2017), Havasi et al. (2018) sacrifice the analytical tractability of the $F_M$ (having only access to samples) in order to allow for the dependence between the inducing outputs of different layers $F_M^l$ and also to allow for non-Gaussianity of the distribution over all the $F_M$. Exactly this trade-off is what we investigate further in Chap. 3 of this thesis.

## 2.4 Technical Background on Gaussian Process State-Space Models

As we discussed in Sec. 1.2.3, GP regression is in principle applicable to time series modelling problems, but only suboptimally so since it disregards the important role that time plays in these problems. Gaussian process state-space models provide a flexible and principled framework capable of dealing with time series in a fully probabilistic manner. While we discussed the idea behind GPSSMs and the difference between state-of-the-art approaches in Sec. 1.4, here we also provide a short overview over other works dealing with inference in GPSSMs, and the necessary technical background. As we will see, all recent GPSSM inference methods build on the sparse GP frameworks that we discussed in Sec. 2.2. Additionally, recent methods have many similarities to the state-of-the-art deep GP inference approaches introduced in the previous section.

The idea of using GPs to model transitions in state-space models goes back to Wang et al. (2005):

The authors propose to model a time series $Y_T = \{y_t\}_{t=1}^T$ by introducing a latent temporal state $X_{T_0} = \{x_t\}_{t=0}^T$ (including an initial latent state $x_0$)[11] that separates the true dynamics from the observations using an emission model $p(y_t|x_t)$ which allows modelling measurement noise.[12] The latent states are assumed to evolve according to the Markovian transition model $p(x_t|f_{t-1}, x_{t-1})$, where $f_{t-1} \equiv f(x_{t-1})$ are the evaluations of a transition function $f$ on which we place a GP prior with kernel $k$ leading to $p(F_T|X_T) = \mathcal{N}(F_T|0, K_{TT})$, where $F_T = \{f_t\}_{t=0}^{T-1}$ and $K_{TT} = \{k(x_t, x_{t'})\}_{t,t'=0}^{T-1}$ [cf. Eq. (1.1)]. Additionally specifying an initial distribution $p(x_0)$ leads to the joint prior model

$$p(Y_T, X_{T_0}, F_T) = p(x_0) \prod_{t=1}^{T} p(y_t|x_t) p(x_t|f_{t-1}, x_{t-1}) p(f_{t-1}|f_{0:t-2}, x_{0:t-1}), \tag{2.24}$$

where we have factorised $p(F_T|X_T) = \prod_{t=1}^T p(f_{t-1}|f_{0:t-2}, x_{0:t-1})$ using the notation $x_{0:t-1} = \{x_{t'}\}_{t'=0}^{t-1}$ and the individual factors can be obtained using Eq. (A.3) recursively. A typical choice for the different parts of the model is given by

$$p(y_t|x_t) = \mathcal{N}\left(y_t \middle| Ax_t + b, \sigma_y^2\right), \quad p(x_t|f_{t-1}, x_{t-1}) = \mathcal{N}\left(x_t \middle| f_{t-1}, \sigma_x^2\right), \quad p(x_0) = \mathcal{N}(x_0|0, 1), \tag{2.25}$$

where we introduced model parameters $\{A, b, \sigma_y, \sigma_x\}$.[13]

Training the GP in this model is an inherently hard task since the GP simultaneously needs to fulfil two (possibly opposing) tasks: On one hand, the function $f$ needs to provide a good *output* mapping for $x_t$ such that the emission model $p(y_t|x_t)$ has an easy task to model the outputs. On the other hand, the GP also needs to provide a good *input* mapping for $x_t$ such that this can be used as the input for modelling the next state $x_{t+1}$ through the transition model. Learning in the model given by Eqs. (2.24) and (2.25) amounts to finding settings for the model parameters and the GP hyperparameters as well as performing inference over the latent variables $\{X_{T_0}, F_T\}$. In the work by Wang et al. (2005), learning was performed by finding a maximum a posteriori estimate of the latent variables and the other parameters.

The first Bayesian treatment of the latent states and the transition function in GPSSMs can be found in Frigola et al. (2013) where a Markov chain Monte Carlo approach was used. The authors provide two versions, one with a standard GP and one using the FITC approach to sparse GPs [Eq. (2.10)] in order to reduce the computational complexity. However, the runtime of these approaches remained high, mainly due to the Markov chain Monte Carlo computations such that later works focused on variational approximations, beginning with Frigola et al. (2014): They used a sparse GP in the flavour of Titsias (2009), introducing inducing variables $X_M, F_M$ to the GPSSM model [Eq. (2.24)] in order to deal with the GP in the transition function. This leads to the augmented joint model [cf. the plate

[11] Note that the subindex 0 in our notation $X_{T_0}$ denotes the inclusion of $x_0$ in the set $X_T = \{x_t\}_{t=1}^T$, i.e., $X_{T_0} = X_T \cup \{x_0\}$.

[12] Additionally, this allows using a latent state $x_t$ with a higher dimensionality than the observations $y_t$, which is e.g. required when modelling a position using the latent velocity and acceleration. For GPSSMs a high dimensional latent state is typically additionally required to reduce non-identifiabilities (see footnote 13).

[13] While early approaches typically considered much more expressive emission models $p(y_t|x_t)$ than the one used here, it has been shown that the linear model in Eq. (2.25) is as powerful as choosing a GP transition *and* emission model, provided the dimensionality of the latent state $x_t$ is high enough (Frigola, 2015, Sec. 3.2.1). The current parametrisation reduces non-identifiabilities, i.e., helps attributing patterns in the data to either the dynamics or to the static observation model. Furthermore, choosing a standard normal distribution over $x_0$ is justified when the data is normalised.

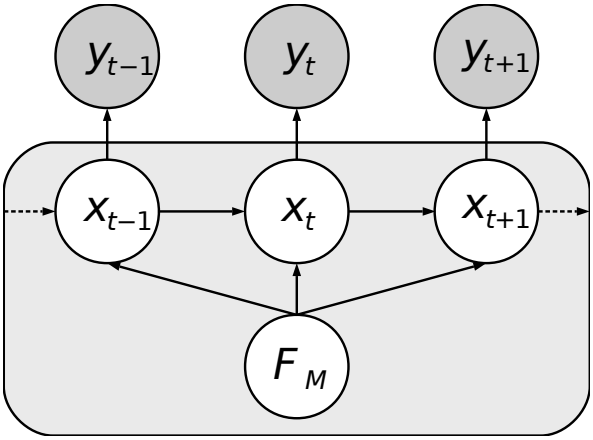


**Figure 2.3:** Plate diagram for the GPSSM using the FITC approximation: The observations $y_t$ depend only on the corresponding temporal latent state $x_t$. Those are influenced by the previous latent state $x_{t-1}$ which are conditioned on the inducing outputs $F_M$.

diagram in Fig. 1.4]]

$$p(Y_T, X_{T_0}, F_T, F_M) = p(x_0)p(F_M)\prod_{t=1}^{T} p(y_t|x_t)p(x_t|f_{t-1}, x_{t-1})p(f_{t-1}|f_{0:t-2}, x_{0:t-1}, F_M), \tag{2.26}$$

where $\prod_{t=1}^{T} p(f_{t-1}|f_{0:t-2}, x_{0:t-1}, F_M) = p(F_T|X_T, F_M)$ is as in Eq. $(2.7)$ with $N$ replaced by $T$. For the model in Eq. $(2.26)$, Frigola et al. (2014) proposed the variational family $q(X_{T_0}, F_T, F_M) = q(X_{T_0})q(F_M)p(F_T|X_T, F_M)$ such that the expensive term $p(F_T|X_T, F_M)$ cancels in the ELBO while the other two distributions are free form. Due to the independence assumption between $F_M$ and $X_{T_0}$, the form of the optimal distributions $q(X_{T_0})$ and $q(F_M)$ can be obtained, where $q(X_{T_0})$ is analytically intractable. Frigola et al. (2014) use a particle filtering approach to draw samples from that distribution. In a later work, Eleftheriadis et al. (2017) improved the efficiency of this method by using a doubly stochastic variational inference scheme that allows for the first time for mini-batches. They opt for a parametric Gaussian distribution for $q(F_M)$ and a linear Markov Gaussian model for the temporal latent states, $q(X_{T_0}) = q(x_0)\prod_{t=1}^{T} q(x_t|x_{t-1})$, where $q(x_t|x_{t-1}) = \mathcal{N}(x_t|A_t x_{t-1}, B_t)$ with parameters $A_t$ and $B_t$ for every time step. The authors additionally employ a recognition model to amortise the inference of the many parameters of such an approach.

Only recently have variational methods also incorporated the dependency between the $F_M$ and the $X_{T_0}$ in their approximations: The method by Doerr et al. (2018) called probabilistic recurrent state-space model (PRSSM) employs the FITC approximation [Eq. $(2.10)$][14] to simplify the prior in Eq. $(2.26)$ by marginalising the $F_T$, leading to

$$p(Y_T, X_{T_0}, F_M) = p(x_0)p(F_M)\prod_{t=1}^{T} p(y_t|x_t)p(x_t|x_{t-1}, F_M), \tag{2.27}$$

$$p(x_t|x_{t-1}, F_M) = \mathcal{N}\left(x_t \middle| K_{t-1,M}K_{MM}^{-1}F_M, \sigma_x^2 + k_{t-1,t-1} - K_{t-1,M}K_{MM}^{-1}K_{t-1,M}^{\top}\right) \tag{2.28}$$

[14] Note that while the reason for using the FITC approximation is the same in both cases (intractable scaling, here with $\mathcal{O}(T^3)$, in standard GP regression with $\mathcal{O}(N^3)$), there are some differences in the two cases: For the GP regression case, the inducing outputs $F_M$ can be efficiently analytically marginalised which is not the case for the GPSSM. Therefore, an approximate posterior also over the $F_M$ is required here [see Eq. (2.29)]. The latter resolves the issue of vanishing noise variances that the FITC approximation encounters in some cases (see Fig. 2.2, top).

which we derive in Appx. A.2.3. The plate diagram for this model can be found in Fig. 2.3. For this model the authors propose to use the variational family

$$q(X_{T_0}, F_M) = q(x_0)q(F_M)\prod_{t=1}^{T} p(x_t|x_{t-1}, F_M), \tag{2.29}$$

i.e., they respect the conditional dependencies between the $X_{T_0}$ and the $F_M$ but use the simplest possible model for $q(X_T|F_M)$, namely the prior. The variational distribution in Eq. $(2.29)$ is inspired by the similar one for the deep GP by Salimbeni and Deisenroth (2017) [cf. Eq. $(2.19)$] and consequently leads to a structurally similar ELBO [cf. Eq. $(2.20)$],

$$\mathcal{L}_{\text{PRSSM}} = \sum_{t=1}^{T} \mathbb{E}_{q(x_t)}\left[\log p(y_t|x_t)\right] - \text{KL}\left[q(x_0) \parallel p(x_0)\right] - \text{KL}\left[q(F_M) \parallel p(F_M)\right], \tag{2.30}$$

$$q(x_t) = \iint \left[\prod_{t'=1}^{t} p(x_{t'}|x_{t'-1}, F_M)\right] q(x_0)q(F_M)dF_M dx_{0:t-1}, \tag{2.31}$$

which we also derive in Appx. A.2.3. While the KL-divergences in Eq. $(2.30)$ are analytically tractable when all involved distributions are Gaussian, the marginals $q(x_t)$ in Eq. $(2.31)$ are intractable and have to be approximated. Doerr et al. (2018) propose to approximate the integration over $F_M$ by changing the order of the integral and the product,

$$q(x_t) \approx \int \left[\prod_{t'=1}^{t} \int p(x_{t'}|x_{t'-1}, F_M)q(F_M)dF_M\right] q(x_0)dx_{0:t-1} \tag{2.32}$$

$$= \int \left[\prod_{t'=1}^{t} q(x_{t'}|x_{t'-1})\right] q(x_0)dx_{0:t-1}, \tag{2.33}$$

where $q(x_{t'}|x_{t'-1})$ is analytically tractable. However, this approximation effectively leads to having independent GP samples for every time step $t$ (see also the discussion in Ialongo et al., 2019, Sec. 4.7; and Longi et al., 2021, Appx. A.2). Furthermore, the resulting expression in Eq. $(2.33)$ is still not analytically tractable since the latent states $x_{t'}$ appear as inputs to kernels. This is exactly the same problem that the deep GP inference scheme by Salimbeni and Deisenroth (2017) faced [cf. Eq. $(2.23)$] and that is consequently solved in the same way, i.e., by recursively sampling the temporal latent states. This is the first source of stochasticity in the algorithm of Doerr et al. (2018). The second source of stochasticity comes from the fact that the ELBO in Eq. $(2.30)$ is a sum over the observations $y_t$ and is therefore amenable to mini-batching (or rather using subsequences of $Y_T$),[15] making the method of Doerr et al. (2018) also a doubly stochastic variational inference approach, similarly as the method of Salimbeni and Deisenroth (2017) for deep GPs.

In a follow-up work, Ialongo et al. (2019) criticise several aspects of the approach by Doerr et al. (2018), namely i) the inexpressive choice of the prior[16] as the approximate smoothing distribution $q(X_{T_0}|F_M)$ in Eq.$(2.29)$, ii) the FITC approximation leading to the approximate model in Eq. $(2.27)$, and finally iii) the approximation of the marginal $q(x_t)$ in Eq. $(2.32)$. The authors therefore propose

[15] Note that naively using subsequences leads to a crude approximation since this disregards the temporal dependencies throughout the whole sequence $Y_T$. See also the extensive discussion in Aicher et al. (2019).

[16] Choosing the prior as the approximate posterior is generally an extremely inexpressive choice in variational inference: Basically, this amounts to fully relying on the prior modelling assumptions since the data cannot influence the posterior model in this case.

the following method: They use the full model in Eq. (2.26) instead of Eq. (2.27), thus remedying ii) and propose the following variational family:

$$q(X_{T_0}, F_T, F_M) = q(x_0)p(F_T|X_T, F_M)q(F_M)\prod_{t=0}^{T-1} q(x_{t+1}|F_M, x_t), \tag{2.34}$$

$$q(x_{t+1}|F_M, x_t) = \mathcal{N}\left(x_{t+1}\middle|A_t K_{tM}K_{MM}^{-1}F_M + b_t, S_t + A_t\left[k_{tt} - K_{tM}K_{MM}^{-1}K_{tM}^\top\right]A_t^\top\right). \tag{2.35}$$

The main difference to the variational family of Doerr et al. (2018) in Eq. (2.29) lies in the replacement of the prior term in Eq. (2.28) through the term in Eq. (2.35) which is a fix for criticism i). The choice of the latter term is inspired by the true posterior filtering factors $p(x_{t+1}|f_{0:t}, x_t, y_{1:t+1})$ that have the same functional form, i.e., a non-linear Markov Gaussian model. The parameterisation in Eq. (2.35) is an approximation to that form that allows inference in $\mathcal{O}(T)$ time but introduces new parameters $A_t$, $b_t$, and $S_t$ for every time step $t$, which additionally have to be inferred. Note that by setting $A_t = 1$, $b_t = 0$, and $S_t = \sigma_x^2$, the prior transition model in Eq. (2.28) is recovered. Hence, the PRSSM method by Doerr et al. (2018) is a special case of the approach by Ialongo et al. (2019) which the authors named variationally coupled dynamics and trajectories (VCDT). The ELBO for this method can be obtained by plugging the prior model [Eq. (2.26)] and the approximate posterior [Eq. (2.34)] in Eq. (A.12), the general formula for the ELBO, yielding

$$\mathcal{L}_{\text{VCDT}} = \mathcal{L}_{\text{PRSSM}} - \sum_{t=1}^{T}\mathbb{E}_{q(x_{t-1}, f_{t-1}, F_M)}\left[\text{KL}\left[q(x_t|F_M, x_{t-1}) \parallel p(x_t|f_{t-1}, x_{t-1})\right]\right]. \tag{2.36}$$

Here $\mathcal{L}_{\text{PRSSM}}$ is the ELBO of the PRSSM approach in Eq. (2.30) and the additional term comes from replacing the prior transition in Eq. (2.28) by the new parametric form in Eq. (2.35). We derive Eq. (2.36) in detail in Appx. A.2.4, providing also the formulas for the marginals $q(x_t)$ [cf. Eq. (2.31)] and $q(x_{t-1}, f_{t-1}, F_M)$ there. When it comes to optimising the ELBO, Ialongo et al. (2019) face the same problem as Doerr et al. (2018): While the KL-terms are analytically tractable, the marginals $q(x_t)$ and $q(x_{t-1}, f_{t-1}, F_M)$ [see Eqs. (A.30) and (A.32)] are not. Instead of using the biased approximation in Eq. (2.32), Ialongo et al. (2019) propose to sample from $q(F_M)$ instead which yields an unbiased estimator (at the price of a higher variance). This therefore constitutes a solution to the final criticism iii) that Ialongo et al. (2019) stated about the method of Doerr et al. (2018).

In summary, Ialongo et al. (2019) improve the method of Doerr et al. (2018) by using a much more expressive variational family (that even contains the less expressive family as a special case) and by removing two approximating assumptions. It is therefore quite surprising that Ialongo et al. (2019) find in their experiments that in many cases the easier model still outperforms their more powerful one.

This raises several questions that we address throughout the thesis: We will first investigate the role of the two approximations, the FITC assumption and the one made in Eq. (2.32), more closely and thereby get a better understanding of GPSSMs in Chap. 4. We will then use this gained knowledge in Chap. 5 to come up with a new method that is specially designed for time series consisting of slowly and quickly changing components. Our method is based on the one by Doerr et al. (2018) and outperforms other approaches on the specific task it has been designed for, but also generally improves the efficiency of GPSSMs. Finally, in Chap. 6, we will come back to the performance issues of Ialongo et al. (2019) and design a new GPSSM. Our approach trades off the expressivity of the VCDT method by Ialongo et al. (2019) with the good optimisation properties of the PRSSM approach by Doerr et al. (2018), thus arriving at a method which can outperform both.

CHAPTER 3

# Structured Deep Gaussian Processes

Gaussian processes (GPs, see Sec. 2.1) provide a non-parametric framework for learning distributions over unknown functions from data (Rasmussen and Williams, 2006): As the posterior distribution can be computed in closed-form, they return well-calibrated uncertainty estimates, making them particularly useful in safety critical applications (Amodei et al., 2016; Reeb et al., 2018), Bayesian optimisation (Snoek et al., 2012; Hebbal et al., 2021), active learning (Zimmer et al., 2018) or under covariate shift (Snoek et al., 2019). However, the analytical tractability of GPs comes at the price of reduced flexibility: Standard kernel functions make strong assumptions such as stationarity or smoothness. To make GPs more flexible, a practitioner would have to come up with hand-crafted features or kernel functions. Both alternatives require expert knowledge and are prone to overfitting (see Sec. 1.2.2).

Deep Gaussian processes (see Sec. 2.3) offer a compelling alternative since they learn non-linear feature representations in a fully probabilistic manner via GP cascades (Damianou and Lawrence, 2013). The gained flexibility has the drawback that inference can no longer be carried out in closed-form, but must be performed via Monte Carlo sampling (Havasi et al., 2018), or approximate inference techniques (Damianou and Lawrence, 2013; Bui et al., 2016; Salimbeni and Deisenroth, 2017). The most popular approximation, variational inference (see Sec. A.2), searches for the best approximate posterior within a pre-defined class of distributions: the variational family (Blei et al., 2017). For GPs, variational approximations often build on the inducing point framework (see Sec. 2.2) where a small set of latent variables, the set of inducing outputs $F_M$, acts as pseudo data points summarising the training data (Snelson and Ghahramani, 2005). For deep GPs, each latent GP is governed by its own set of inducing variables, which, in general, need not be independent from those of other latent GPs.

The state-of-the-art approaches by Salimbeni and Deisenroth (2017) and Havasi et al. (2018) (that we discuss in detail in Sec. 2.3) offer an interesting trade-off for inference in deep GPs where the main difference lies in the treatment of the inducing outputs $F_M$. Havasi et al. (2018) propose a maximally expressive treatment of the $F_M$ that is advantageous for well calibrated uncertainty estimates (Turner and Sahani, 2011). However, this has the downside that the $F_M$ cannot be treated analytically and are only available as samples which leads to higher variance and therefore generally worse convergence properties during training. In contrast, the method by Salimbeni and Deisenroth (2017) proposes a rather simple approach, i.e., assuming the $F_M$ to be Gaussian distributed and independent between and within different layers of the deep GP. This renders the $F_M$ analytically tractable which reduces the variance in the estimators and is needed for fast convergence (Kingma et al., 2015).

In the current chapter we explore this trade-off further by taking the best of both worlds and offering a new class of variational families for deep GPs (Lindinger et al., 2020). We require that (i) all global latent variables, i.e., inducing outputs $F_M$ can be marginalised out, and that (ii) correlations between latent GP models can be captured. We propose using a fully-parameterised Gaussian variational posterior over the global latent variables, i.e., allowing for correlations between the inducing outputs within and across layers, thus automatically fulfilling (ii). In Sec. 3.1, we show via a proof by induction, that (i) can still be achieved. The proof is constructive, resulting in a novel inference

scheme for variational families that allow for correlations within and across layers. The proposed scheme is general and can be used for arbitrarily structured covariances.[1] In the remainder of Sec. 3.1 we empirically explore the properties of our proposed method. Among other things we find an interesting structure in the covariance matrices governing the correlation of the inducing outputs within and between different layers, pointing towards more and less important correlations. In Sec. 3.2, we further propose a scalable approximation to this variational family, which only takes the stronger correlations into account. We provide efficient implementations for both variational families, where we particularly exploit the sparsity and structure of the covariance matrix of the variational posterior. We then further show experimentally that our new algorithm works well in practice: Our approach obtains a better balance between accurate predictions and calibrated uncertainty estimates than its competitors, as we showcase by varying the distance of the test from the training points (Lindinger et al., 2020). Finally, in Sec. 3.3 we summarise and conclude the current chapter.

## 3.1 Fully-Coupled Deep Gaussian Processes

We laid the theoretical groundwork for the treatment of variational deep GPs in Sec. 2.3, where we introduced the prior model and the conventional variational family in Eqs. $(2.18)$ and $(2.19)$, and the resulting ELBO which depends on the last layer marginals $q(f_n^L)$ in Eqs. $(2.20)$ and $(2.21)$, respectively. Due to the importance of the $q(f_n^L)$ for our current undertaking, we repeat the general formula here:

$$q(f_n^L) = \int \left[ \int q(F_M) \prod_{l=1}^{L} p(f_n^l | F_M^l, f_n^{l-1}) dF_M \right] df_n^1 \cdots df_n^{L-1}, \tag{3.1}$$

where the $p(f_n^l | F_M^l, f_n^{l-1})$ are given in Eq. $(\mathrm{A}.21)$ and $q(F_M)$ is the part of the variational family [Eq. $(2.19)$] describing the inducing outputs. We saw that the mean-field variational family $q_{\mathrm{MF}}(F_M) = \prod_{l=1}^{L} \prod_{t=1}^{T_l} \mathcal{N}\left(F_M^{l,t} \middle| \mu_M^{l,t}, S_M^{l,t}\right)$ [Eq. $(2.22)$] by Salimbeni and Deisenroth (2017) (which ignores correlations of GPs between and within different layers) leads to the inner integral in Eq. $(3.1)$ being analytically tractable [see Eq. $(2.23)$ and Appx. A.2.2].

Instead, we propose a new variational family that allows to couple the inducing outputs but retains the analytical marginalisation property. We do this by leaving the Gaussianity assumption unchanged, while permitting dependencies between all inducing outputs (within layers and also across layers). This corresponds to the fully-coupled (FC) variational ansatz

$$q_{\mathrm{FC}}(F_M) = \mathcal{N}(F_M | \mu_M, S_M) \tag{3.2}$$

with dimensionality $TM$, where $T$ is the total number of GPs in the deep GP and $M$ is the number of inducing points (per GP). By taking the dependencies between the latent processes into account, the resulting variational posterior $q(F_N, F_M)$ [Eq. $(2.19)$] is more expressive, i.e., better suited to closely approximate the true posterior and contains the mean-field variational family of Salimbeni and Deisenroth (2017) as a special case. As we will see next, this approach is also computationally efficient since the inducing outputs can still be marginalised out.

[1] One particular case, in which the variational family is chain-structured, has also been considered in a recent work in which the compositional uncertainty in deep GP models is studied (Ustyuzhaninov et al., 2020b).

### 3.1.1 Analytical Marginalisation of the Inducing Outputs

Exchanging the distribution $q_{\text{MF}}(F_M)$ by $q_{\text{FC}}(F_M)$ has no influence on the general form of the ELBO for deep GPs in Eq. (2.20). Only when it comes to the last layer marginals in Eq. (3.1), does the new variational family make a difference: $q_{\text{MF}}(F_M)$ can be written as a product over the $L$ different layers and therefore reduces the inner integral in Eq. (3.1) to $L$ independent integrals which can be solved using standard Gaussian calculus (see Appx. A.2.2). This is not the case for the fully-coupled deep GP which makes the computations more challenging. The implications of using a fully coupled $q_{\text{FC}}(F_M)$ [Eq. (3.2)] are summarised in the following theorem (Lindinger et al., 2020).

**Theorem 3.1.** *In a fully-coupled deep GP, the last layer marginals $q(f_n^L)$ can be written as*

$$q_{FC}(f_n^L) = \int \prod_{l=1}^{L} q(f_n^l | f_n^{1:l-1}) df_n^1 \cdots df_n^{L-1}, \quad \textit{where} \quad q(f_n^l | f_n^{1:l-1}) = \mathcal{N}\left(f_n^l \middle| \hat{\mu}_n^l, \hat{\Sigma}_n^l\right), \tag{3.3}$$

*for each data point $x_n$. The means and covariances are given by*

$$\hat{\mu}_n^l = \widetilde{\mu}_n^l + \widetilde{S}_n^{l,1:l-1} \left(\widetilde{S}_n^{1:l-1,1:l-1}\right)^{-1} (f_n^{1:l-1} - \widetilde{\mu}_n^{1:l-1}), \tag{3.4}$$

$$\hat{\Sigma}_n^l = \widetilde{S}_n^{ll} - \widetilde{S}_n^{l,1:l-1} \left(\widetilde{S}_n^{1:l-1,1:l-1}\right)^{-1} \widetilde{S}_n^{1:l-1,l}, \tag{3.5}$$

*where we introduced the shorthand notations*

$$\begin{aligned} \widetilde{\mu}_n^l &= \mathcal{K}_{nM}^l \left(\mathcal{K}_{MM}^l\right)^{-1} \mu_M^l \\ \widetilde{S}_n^{ll'} &= \delta_{ll'} \mathcal{K}_{nn}^l - \mathcal{K}_{nM}^l \left(\mathcal{K}_{MM}^l\right)^{-1} \left(\delta_{ll'} \mathcal{K}_{MM}^l - S_M^{ll'}\right) \left(\mathcal{K}_{MM}^{l'}\right)^{-1} \left(\mathcal{K}_{nM}^{l'}\right)^\top. \end{aligned} \tag{3.6}$$

In Eqs. (3.4) and (3.5) we use e.g. $A^{l,1:l'} = \left(A^{l,1} \cdots A^{l,l'}\right)$ to denote a submatrix of the variable $A$.[2] Additionally, $\mu_M^l \in \mathbb{R}^{T_l M}$ denotes the sub-vector of $\mu_M$ that contains the means of the inducing outputs in layer $l$, and $S_M^{ll'} \in \mathbb{R}^{T_l M \times T_{l'} M}$ contains the covariances between the inducing outputs of layers $l$ and $l'$. In Eq. (3.6), we introduced the notation $\mathcal{K}^l = \left(I_{T_l} \otimes K^l\right)$ as shorthand for the Kronecker product between the identity matrix $I_{T_l}$ and the covariance matrix $K^l$,[3] and used $\delta$ for the Kronecker delta.

By Thm. 3.1, the inducing outputs $F_M$ can still be marginalised out, which enables low-variance estimators of the ELBO. While the resulting formula for $q(f_n^l | f_n^{1:l-1})$ has a similar form as Gaussian conditionals, this is only true at first glance: The latents of the preceding layers $f_n^{1:l-1}$ enter the mean $\hat{\mu}_n^l$ and the covariance matrix $\hat{\Sigma}_n^l$ also in an indirect way via $\widetilde{S}_n$ as they appear as inputs to the kernel matrices.

**Sketch of the proof of Theorem 3.1**. We start the proof with Eq. (3.1), the general formula for the last layer marginals $q(f_n^L)$. In order to show the equivalence between the inner integral in Eq. (3.1) when plugging in Eq. (3.2) and the integrand in Eq. (3.3) we proceed to find a recursive formula for

[2] Note that these formulas contain the mean-field solution in Eq. (2.23) as a special case. This can be seen by plugging in the respective covariance matrix which is done in Lindinger et al. (2020, Appx. B.2).

[3] We follow Salimbeni and Deisenroth (2017) in choosing the same kernel $k^l$ and the same inducing inputs $X_M^l$ for every GP in the $l$-th layer. This leads to $K_{nM}^{l,t} = K_{nM}^l$ and $K_{MM}^{l,t} = K_{MM}^l \forall t$ which allows us to use the Kronecker product.

integrating out the inducing outputs layer after layer:

$$\begin{aligned}&\int q_{\text{FC}}(F_M)\prod_{l=1}^{L} p(f_n^l|F_M^l, f_n^{l-1})dF_M \\ &= \left[\prod_{l'=1}^{l-1} q(f_n^{l'}|f_n^{1:l'-1})\right]\int q(f_n^l, F_M^{l+1:L}|f_n^{1:l-1})\prod_{l'=l+1}^{L} p(f_n^{l'}|F_M^{l'}, f_n^{l'-1})dF_M^{l'}.\end{aligned} \tag{3.7}$$

The equation above holds for $l = 1, \ldots, L$ after the inducing outputs of layers $1, \ldots, l$ have already been marginalised out. This is stated more formally in Lem. B.1 in Appx. B.1, in which we also provide exact formulas for all terms. Importantly, all of them are multivariate Gaussians with known mean and covariance. The lemma itself can be proved by induction and we will show the general idea of the induction step here: For this, we assume the right hand side (RHS) of Eq. (3.7) to hold for some layer $l$ and then prove that it also holds for $l \to l+1$. We start by taking the (known) distribution within the integral and split it in two by conditioning on $f_n^l$:

$$q(f_n^l, F_M^{l+1:L}|f_n^{1:l-1}) = q(f_n^l|f_n^{1:l-1})q(F_M^{l+1:L}|f_n^{1:l}) \tag{3.8}$$

Then we show that the distribution $q(f_n^l|f_n^{1:l-1})$ can be written as part of the product in front of the integral on the RHS of Eq. (3.7) (thereby increasing the upper limit of the product to $l$). Next, we consider the integration over $F_M^{l+1}$, where we collect all relevant terms [thereby increasing the lower limit of the product within the integral on the RHS of Eq. (3.7) to $l+2$]:

$$\begin{aligned}&\int q(F_M^{l+1:L}|f_n^{1:l})p(f_n^{l+1}|F_M^{l+1}, f_n^l)dF_M^{l+1} \\ &= \int q(F_M^{l+1}|f_n^{1:l})q(F_M^{l+2:L}|f_n^{1:l}, F_M^{l+1})p(f_n^{l+1}|F_M^{l+1}, f_n^l)dF_M^{l+1} \\ &= \int q(F_M^{l+1}|f_n^{1:l})q(f_n^{l+1}, F_M^{l+2:L}|f_n^{1:l}, F_M^{l+1})dF_M^{l+1} = q(f_n^{l+1}, F_M^{l+2:L}|f_n^{1:l}).\end{aligned} \tag{3.9}$$

The terms in the first line are given by Eqs. (3.8) and (A.21). All subsequent terms are also multivariate Gaussians that are obtained by standard operations like conditioning, joining two distributions, and marginalisation (see e.g. Toussaint, 2011). We can therefore give an analytical expression of the final term in Eq. (3.9), which is exactly the term that is needed on the right hand side of Eq. (3.7) for $l \to l+1$. Confirming that this term has the correct mean and covariance completes the induction step.

After proving Lem. B.1, Eq. (3.7) can be used. For the case $l = L$ the right hand side can be shown to yield $\prod_{l=1}^{L} q(f_n^l|f_n^{1:l-1})$. Hence, Eq. (3.3) follows by substituting the inner integral in Eq. (3.1) by this term. The full proof can be found in Appx. B.1. □

### 3.1.2 Experiments

We use our novel variational approach to fit fully-coupled deep GP models with different architectures to some UCI regression datasets.[4] We vary the number of layers $L$ and the number of GPs per latent layer which we denote as $\tau$. Since the regression data sets have one-dimensional outputs, we always use a single GP in the last layer. The resulting covariance matrices $S_M$ are depicted in Fig. 3.1 where we can clearly observe that our algorithmic work pays off: There is more structure in the

[4] The algorithm and experimental details can be found in Appxs. F and G of Lindinger et al. (2020), respectively. Code is publicly available at https://github.com/boschresearch/Structured_DGP.

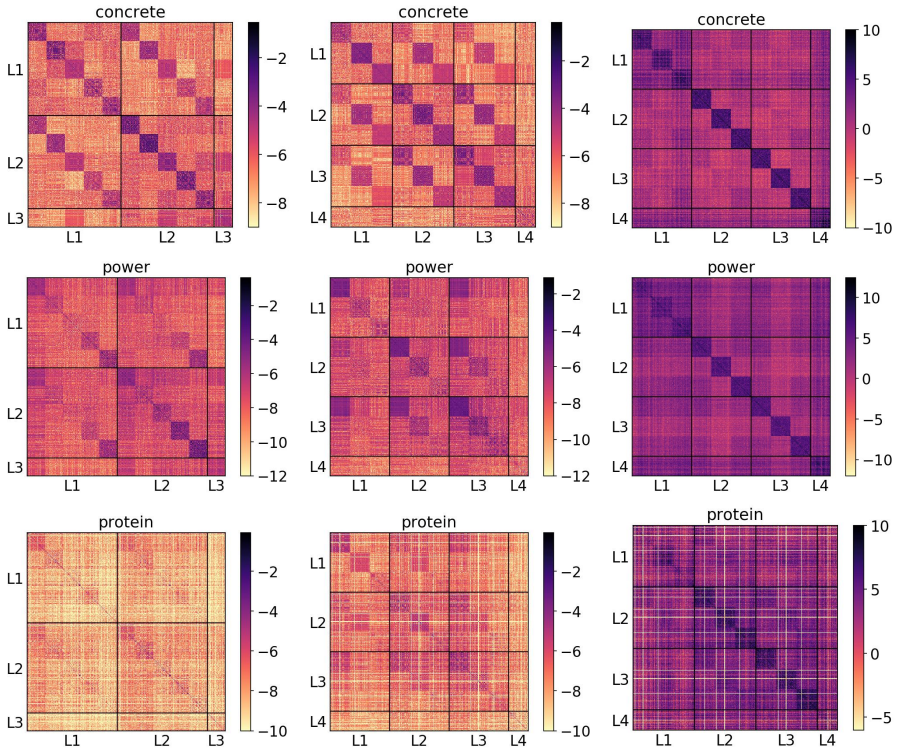


**Figure 3.1:** Covariance/precision matrices after optimisation for three different UCI data sets. The first column depicts covariance matrices $S_M$ for our standard architecture, $L = 3, \tau = 5$, while the second column depicts covariance matrices for $L = 4, \tau = 3$. The third column shows the precision matrices corresponding to the second column, i.e., the inverse matrices. Plotted are natural logarithms of the absolute values of the variational covariance/precision matrices over the inducing outputs.

covariance matrices than the mean-field approximation allows (which amounts to a block-diagonal $S_M$, cf. Eq. $(2.22)$). This can be seen even more clearly in a direct comparison of the two methods in Fig. 3.3 in the following section.

The fact that we uncover additional structure in the covariance matrices $S_M$ implies that the fully-coupled variational family in Eq. $(3.2)$ is a better approximation of the true posterior than the mean-field variational family [Eq. $(2.22)$]. Since our variational family contains the latter one as a special case, this is not surprising but still a valuable sanity test. We further validate this by comparing the values of the ELBO [Eq. $(2.20)$] that both models achieve after the optimisation. The results can be seen in Tab. 3.1 and we find that our fully-coupled approximation yields better ELBOs for all datasets.

These comparisons cover the first part of the trade-off that we set out to achieve by introducing the new variational family: more expressivity than the mean-field approach by Salimbeni and Deisenroth (2017). The second part, a method with better convergence properties than the approach by Havasi et al. (2018) is much harder to experimentally validate. We argued that our method should achieve this since we are able to analytically marginalise the $F_M$ which leads to lower variance estimators of the ELBO, while Havasi et al. (2018) have to sample the $F_M$. A direct experimental validation of this is not possible since the two methods build on very different approaches for which it is hard to find a comparable setting. We therefore perform a proxy experiment in which we compare our analytical marginalisation over the inducing outputs $F_M$ with a method that uses the same variational family but instead approximately marginalises the inducing outputs from Eq. $(3.2)$ via Monte Carlo samples.[5] The comparison in Fig. 3.2 clearly demonstrates that our approach which uses the analytical

[5] Independently from our work, this sampling-based approach has also been proposed in Ustyuzhaninov et al. (2020b) for

**Table 3.1:** We report ELBOs (the larger, the better) for the mean-field (MF) and the fully-coupled (FC) method and the number of data points $N$ as well as the input dimensionality $D$ for each data set. We used our standard architecture with $M = 128$, $\tau = 5$, and $L = 3$ for both methods and trained them using natural gradients (see text). Standard errors are obtained by repeating the experiment 10 times. We warm-started the optimisation of the fully-coupled method from the converged mean-field solution. Significantly better performing methods (non overlapping standard errors) are marked in bold for each dataset.

| Dataset (N,D) | boston (506,13) | energy (768, 8) | concrete (1030, 8) | wine_red (1599,11) | kin8nm (8192, 8) | power (9568, 4) | naval (11934,16) | protein (45730, 9) |
|---|---|---|---|---|---|---|---|---|
| MF | -510(30) | 510(8) | -910(50) | -1648(8) | -2000(100) | -390(90) | 33000(600) | -44040(140) |
| FC | **-246(5)** | **600(20)** | **-500(10)** | **-1575(4)** | **-1290(70)** | **-10(50)** | **34600(500)** | **-42610(120)** |

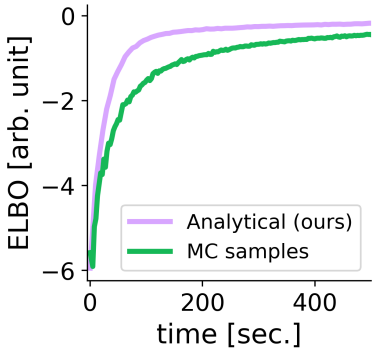


**Figure 3.2:** We plot the ELBO as a function of time in seconds when the marginalisation of the inducing outputs $F_M$ is performed analytically via our Thm. 3.1 (purple) and via Monte Carlo sampling (green). Both curves show results for a fully-coupled deep GP with our standard three layer architecture ($M = 128, L = 3, \tau = 5$), on the *concrete* UCI dataset trained with Adam (Kingma and Ba, 2015).

marginalisation has superior convergence properties. This is even true when measured in total run time where the more complicated computations that have to be performed in our approach are taken into account. We can also see that the optimisation curve for the sampling method is noisier, which can be explained by the higher variance estimates that are to be expected when Monte Carlo sampling is used.

However, in comparison with the mean-field deep GP, the increase in the number of variational parameters also leads to an increase in runtime and made convergence with standard optimisers fragile due to many local optima. We were able to circumvent the latter by the use of natural gradients (Amari, 1998), which have been found to work well for (deep) GP models before (Salimbeni et al., 2018; Salimbeni et al., 2019; Adam et al., 2021; Hebbal et al., 2021), but this increases the runtime even further (see Fig. 3.6 in the following section). It is therefore necessary to find a smaller variational family if we want to use our method in large-scale applications.

## 3.2 The Stripes-and-Arrow Approximation

An optimal variational family combines the best of both worlds, i.e., being as efficient as the mean-field deep GP while retaining the most important interactions introduced in the fully-coupled deep GP. We want to emphasise that there are many possible ways of restricting the covariance matrix $S_M$ that

the fully-coupled variational family.

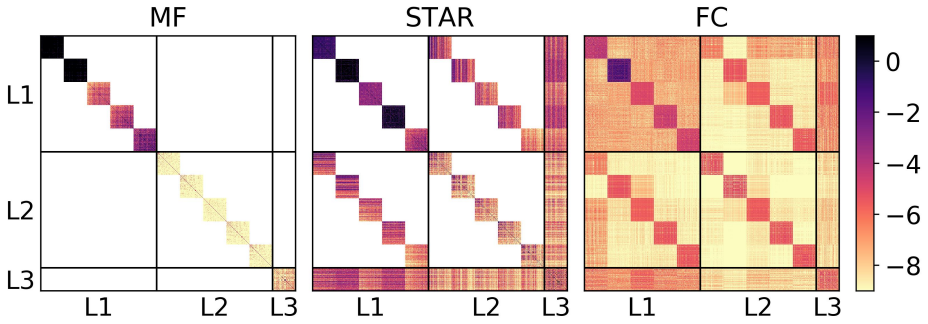


**Figure 3.3:** Depicted are covariance matrices $S_M$ for different variational posteriors. We used a DGP with 2 hidden layers (L1, L2) of 5 latent GPs each and a single GP in the output layer (L3). The complexity of the variational approximation is increased by allowing for additional dependencies within and across layers in a Gaussian variational family (left: mean-field (Salimbeni and Deisenroth, 2017), middle: stripes-and-arrow, right: fully-coupled). Plotted are natural logarithms of the absolute values of the variational covariance matrices over the inducing outputs.

potentially lead to benefits in different applications. For example, the recent work by Ustyuzhaninov et al. (2020b) studies the compositional uncertainty in deep GPs using a particular restriction of the inverse covariance matrix (we depict full inverse covariance matrices for several examples in the rightmost column of Fig. 3.1). The authors also provide specialised algorithms to marginalise out the inducing outputs in their model. In contrast, we provide an analytic marginalisation scheme for *arbitrarily structured* covariance matrices such that a restriction to application-specific covariances (in our case regression on UCI benchmark data sets) is easily achieved. In the following, we propose one possible class of covariance matrices based on our empirical findings that trades off efficiency and expressivity by sparsifying the covariance matrix $S_M$.

Inspecting Figs. 3.1 and 3.3 (right) again, we observe that besides the $M \times M$ blocks on the diagonal, the diagonal stripes (Smolarski, 2006) (covariances between the GPs in latent layers at the same relative position), and an arrow structure (covariances from every intermediate layer GP to the output GP) receive large values. This observation holds for different data sets and different deep GP architectures. Note that the stripes pattern can also be motivated theoretically as we expect the residual connections realised by the mean functions [in order to avoid pathologies created by highly non-injective mappings in the deep GP (Duvenaud et al., 2014), we follow Salimbeni and Deisenroth (2017) and add non-trainable linear mean terms given by the PCA mapping of the input data to the latent layers] to lead to a coupling between successive latent GPs. We therefore propose as one special form to keep only these terms and neglect all other dependencies by setting them to zero in the covariance matrix, resulting in a structure consisting of an arrowhead and diagonal stripes (see Fig. 3.3 middle), which we call stripes-and-arrow approximation.

Denoting the number of GPs per latent layer as $\tau$, it is straightforward to show that the number of non-zero elements in the covariance matrices of mean-field, stripes-and-arrow, and fully-coupled deep GP scale as $\mathcal{O}(\tau L M^2)$, $\mathcal{O}(\tau L^2 M^2)$, and $\mathcal{O}(\tau^2 L^2 M^2)$, respectively. In the example of Fig. 3.3, we have used $\tau = 5$, $L = 3$, and $M = 128$, yielding $1.8 \times 10^5$, $5.1 \times 10^5$, and $2.0 \times 10^6$ non-zero elements in the covariance matrices. Reducing the number of parameters already leads to shorter training times since less gradients need to be computed. Furthermore, the property that makes this form so compelling is that the covariance matrix $\widetilde{S}_n^{1:l-1,1:l-1}$ [needed in Eqs. (3.4) and (3.5)] as well as the Cholesky decomposition[6] of $S_M$ have the same sparsity pattern. Therefore only the non-zero elements at pre-defined positions have to be calculated.

[6] In order to ensure that $S_M$ is positive definite, we will numerically exclusively work with its Cholesky factor $L_C$, a unique lower triangular matrix fulfilling $S_M = L_C L_C^\top$.

In addition to avoiding calculating unnecessary zero blocks there are other calculations that can be sped-up by exploiting properties from linear algebra or by vectorisations. In particular, we can exploit that the calculations for $\widetilde{S}_n^{ll'}$ in Eq. $(3.6)$ involve the $\mathcal{K}^l$ (which are given by Kronecker products) for which efficient matrix multiplication routines exist. Then the calculation of the $\hat{\mu}_n^l$ and $\hat{\Sigma}_n^l$ in Eqs. $(3.4)$ and $(3.5)$ for all $n$ in the data set (or in a batch) can be vectorised which is advantageous especially for computations on GPUs. Finally, for the inversion of $\widetilde{S}_n^{1:l-1,1:l-1}$ in Eqs. $(3.4)$ and $(3.5)$ we can exploit that the matrix is positive semi-definite and therefore possesses a Cholesky decomposition. Instead of directly inverting the matrix for every layer $l$ we will instead calculate the Cholesky factors which can be used for efficient matrix vector solves, and more importantly can be reused to update the Cholesky factor for the next layer. These technical properties and calculations are explained in detail in Lindinger et al. (2020, Appx. E). There, it is also shown that the complexity for the stripes-and-arrow ELBO is $\mathcal{O}(NM^2\tau L^2 + N\tau^3 L^3 + M^3\tau L^3)$. This is a moderate increase compared to the mean-field deep GP whose ELBO has complexity $\mathcal{O}(NM^2\tau L)$, while it is a clear improvement over the fully-coupled approach with complexity $\mathcal{O}(NM^2\tau^2 L^2 + N\tau^3 L^3 + M^3\tau^3 L^3)$.

After having discussed the advantages of the proposed approximation a remark on a disadvantage is in order: The efficient implementation of Salimbeni et al. (2018) for natural gradients cannot be used in this setting, since the transformation from our parameterisation to a fully-parameterised multivariate Gaussian is not invertible. However, this is only a slight disadvantage since the stripes-and-arrow approximation has a drastically reduced number of parameters, compared to the fully-coupled approach, and we experimentally do not observe the same convergence problems when using standard optimisers (see Lindinger et al., 2020, Fig. S5 in Appx. G).

### 3.2.1 Experiments

Next, we study the predictive performance of our stripes-and-arrow approximation. Since it is difficult to assess accuracy and calibration on the same task, we ran a joint study of interpolation and extrapolation tasks, where in the latter the test points are distant from the training points. We compare the predictive performance of our efficient stripes-and-arrow approximation (STAR DGP) with a mean-field approximation (MF DGP, Salimbeni and Deisenroth, 2017) and a stochastic gradient Hamiltonian Monte Carlo approach (SGHMC DGP, Havasi et al., 2018). This allows us to assess the advantages that trading-off the expressivity of the latter method with the good convergence properties of the method by Salimbeni and Deisenroth (2017) bring. As a baseline we also include a sparse GP (SGP, Hensman et al., 2013). As done in prior work, we report results on eight UCI datasets and employ as evaluation criterion the average marginal test log-likelihood.

We first assessed the interpolation behaviour of the different approaches by randomly partitioning the data into a training and a test set with a $90:10$ split, where the results can be found in Lindinger et al. (2020, Tab. 1). This confirmed the reports from the literature (Salimbeni and Deisenroth, 2017; Havasi et al., 2018), that deep GPs have on interpolation tasks an improved performance compared to sparse GPs. We also observed that in this setting SGHMC outperforms the MF DGP and our method, which are on par.

To investigate the extrapolation behaviour, we created test instances that are distant from the training samples: We first randomly projected the inputs $X$ onto a one-dimensional subspace $z = Xw$, where the weights $w \in \mathbb{R}^D$ were drawn from a standard Gaussian distribution. We subsequently ordered the samples w.r.t. $z$ and divided them accordingly into training and test set using a $50:50$ split. Performing the same analysis than on the interpolation task we find that while our approach, STAR DGP, seems to perform slightly better than MF DGP and also SGHMC DGP, the large standard errors of all methods hamper a direct comparison (see Lindinger et al., 2020, Tab. S5 in Appx. G). We attribute this mainly to the random 1D-projection of the extrapolation experiment: The direction of the projection has a large impact on the difficulty of the prediction task. Since this direction changes over

**Table 3.2:** Direct comparison of the different deep GP variants on the extrapolation task. Shown is the average frequency $\mu$ and its standard error $\sigma$ (computed over 10 repetitions) of the STAR DGP outperforming the MF DGP (top row) and the SGHMC DGP (bottom row) on the marginal test log-likelihood of individual repetitions of the extrapolation task (see main text for details). Note that 1 (0) therefore corresponds to our STAR approach always outperforming (being outperformed by) the competitor. Results are for deep GPs with three layers. We mark numbers in bold (italics) if STAR significantly outperforms its competitor (vice versa).

| Dataset | boston | energy | concrete | wine_red | kin8nm | power | naval | protein |
|---|---|---|---|---|---|---|---|---|
| MF vs. STAR | **0.55(0.04)** | **0.73(0.05)** | **0.57(0.04)** | **0.57(0.04)** | *0.36(0.03)* | 0.44(0.06) | **0.67(0.06)** | 0.49(0.03) |
| SGHMC vs. STAR | 0.50(0.05) | **0.60(0.04)** | **0.60(0.03)** | **0.63(0.02)** | *0.44(0.05)* | **0.64(0.03)** | **0.58(0.03)** | 0.50(0.03) |

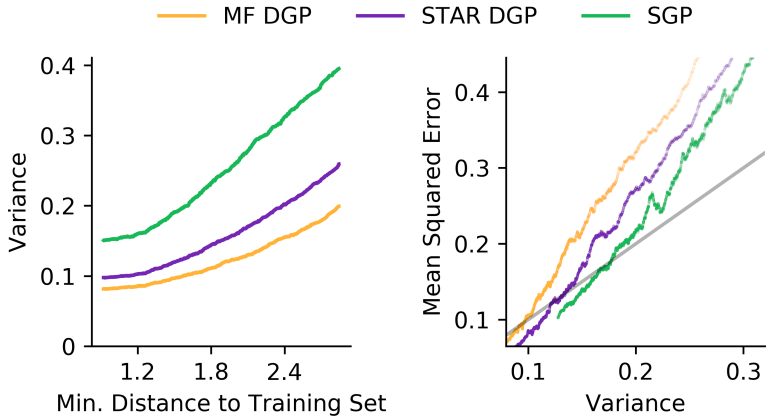


**Figure 3.4:** Depicted are predicted variance and mean squared errors for several methods. The results are recorded on the *kin8nm* UCI dataset and smoothed for plotting by using a median filter. Left: We show the predicted variance as a function of the minimal distance of the test point to the training set. Right: We plot the mean squared error as a function of the predicted variance and compare it to the groundtruth, the identity function shown in grey. Results are recorded on the *kin8nm* UCI dataset and smoothed for plotting by using a median filter.

the repetitions, the corresponding test log-likelihoods vary considerably, leading to large standard errors.

We resolved this issue by performing a direct comparison between STAR DGP and the other two DGP variants: To do so, we computed the frequency of test samples for which STAR DGP obtained a larger log-likelihood than MF/SGHMC DGP on each train-test split independently. Average frequency $\mu$ and its standard error $\sigma$ were subsequently computed over 10 repetitions and are reported in Tab. 3.2. On 5/8 datasets STAR DGP significantly outperforms MF DGP and SGHMC DGP ($\mu > 0.50 + \sigma$), respectively, while the opposite only occurred on *kin8nm*. More comparisons that also take the absolute differences in test log-likelihoods into account and additionally consider the comparison of fully-coupled and MF DGP can be found in Lindinger et al. (2020, Tab. S4). Taken together, we conclude that our structured approximations are in particular beneficial in the extrapolation scenario, while their performance is similar to MF DGP in the interpolation scenario.

Next, we performed an in-depth comparison between the approaches that analytically marginalise the inducing outputs: In Fig. 3.4 we show that the predicted variance $\sigma_*^2$ increased as we moved away from the training data (left) while the mean squared errors also grew with larger $\sigma_*^2$ (right). While the predicted variances increase for all methods as a function of the distance to the training data, we find that at any given distance, the uncertainty decreases from SGP to STAR DGP to MF DGP (Fig. 3.4,

left). The mean squared error is an empirical unbiased estimator of the variance $\text{Var}_* = \mathbb{E}[(y_* - \mu_*)^2]$ where $y_*$ is the test output and $\mu_*$ the mean predictor. The predicted variance $\sigma_*^2$ is also an estimator of $\text{Var}_*$. It is only unbiased if the method is calibrated. However, as Fig. 3.4 (right) shows, we observed for the mean-field approach that, when moving away from the training data, the mean squared error was larger than the predicted variances pointing towards underestimated uncertainties. This is a well-known weakness of choosing a variational family that is too compact (Turner and Sahani, 2011). While the mean squared error for SGP matched well with the predictive variances, the predictions are rather inaccurate as demonstrated by the large predicted variances. Our method, STAR DGP, reaches a good balance, having generally more accurate mean predictions than SGP and at the same time more accurate variance predictions than MF DGP.

To conclude the analysis of the benchmark experiments, we investigated the behaviour of the SGHMC approach in more detail. We ran a one-layer model that is equivalent to a sparse GP but with a different inference scheme: Instead of marginalising out the inducing outputs, they are sampled. We often observed that the distribution over the inducing outputs is non-Gaussian (see Fig. 3.5), even though the optimal approximate posterior distribution is provably Gaussian in this case (Titsias, 2009). A possible explanation for this are convergence problems since the global latent variables are not marginalised out, which, in turn, offers a potential explanation for the poor extrapolation behaviour of SGHMC that we observed in our experiments across different architectures and datasets. Similar convergence problems have also been observed by Salimbeni et al. (2019).

Finally, we compared the impact of the variational family on the runtime as a function of the number of inducing points $M$. The runtime of the different approximations was assessed for a single gradient update averaged over 2,000 updates on a 6 core i7-8700 CPU using a mini-batch size of $512$ and the Adam optimiser (Kingma and Ba, 2015). For the fully-coupled (FC) variational model, we also recorded the runtime when employing natural gradients (Salimbeni et al., 2018). The results can be seen in Fig. 3.6, where the order from fastest to slowest method was proportional to the complexity of the variational family: mean-field, stripes-and-arrow, fully-coupled DGP and then FC DGP trained with natural gradients. For our standard setting, $M = 128$, our STAR approximation was only two times slower than the mean-field but three times faster than FC DGP (trained with Adam). This ratio stayed almost constant when the number of inducing outputs $M$ was changed, since the most important term in the computational costs scales as $\mathcal{O}(M^2)$ for all methods. Additional experiments in which the architecture parameters $L$ and $\tau$ are varied can be found in Lindinger et al. (2020, Fig. S7). They also confirm that the empirical runtime performance scales with the complexity of the variational family and matches our theoretical estimates in Sec. 3.2.

## 3.3 Chapter Summary

In this chapter, we investigated a new class of variational families for deep Gaussian processes (GPs) inspired by achieving a trade-off between the previous state-of-the-art methods. The approach by Salimbeni and Deisenroth (2017) is efficient and has good convergence properties. This is due to the choice of a rather inexpressive approximate posterior, a Gaussian approximation that neglects couplings between different layers, which makes certain global latent variables analytically tractable. In contrast, Havasi et al. (2018) use a maximally expressive free form distribution over the global latent variables which leads to the latter no longer being analytically tractable and therefore to an approach with worse convergence properties. We propose to keep the Gaussianity assumption of Salimbeni and Deisenroth (2017) but allow for arbitrary couplings and show by a proof via induction that with such an approach the global latent variables can still be treated analytically. Further trading off efficiency and expressivity between our newly proposed method that allows for all couplings and the method by Salimbeni and Deisenroth (2017) that ignores couplings between different GPs in the deep GP, we propose another new variational family that takes only the couplings into account that we empirically

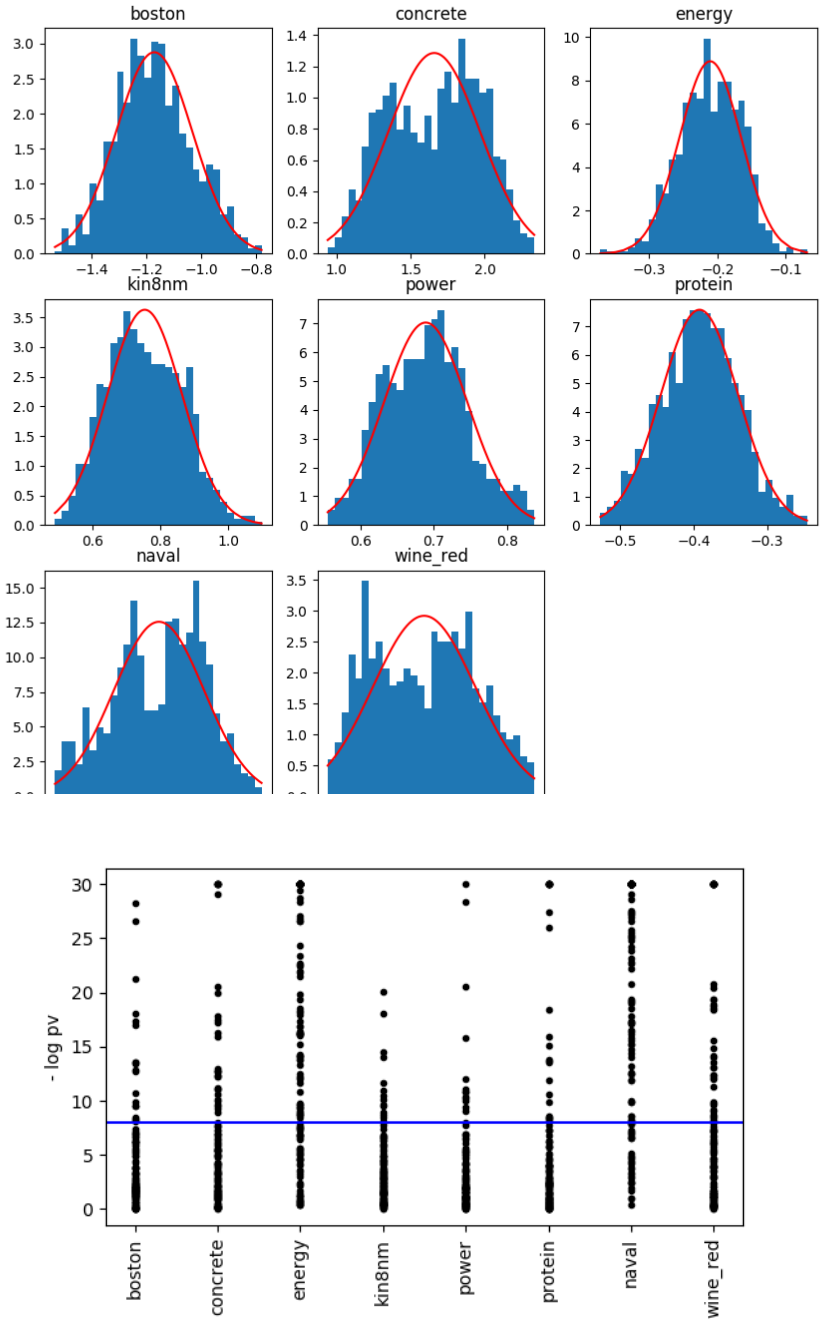


**Figure 3.5:** Depicted is an empirical test whether the marginal distributions $q(f_m)$ over the inducing outputs are Gaussian or not when using SGHMC inference (Havasi et al., 2018). We employ a deep GP with a single layer, equivalent to a sparse GP. Top: Histogram of Monte Carlo samples for a randomly chosen inducing output $f_m$ for all eight benchmark data sets. The red line indicates a Normal distribution fitted to the data. Bottom: For each of the 128 inducing outputs, we computed a p-value if its Monte Carlo samples are normally distributed. We show the negative logarithm of these p-values (the higher this value, the more likely that the samples are not from a Gaussian distribution) for each inducing output and for all data sets. The blue line shows the Bonferroni corrected significance threshold $\alpha = 10^{-5}$.

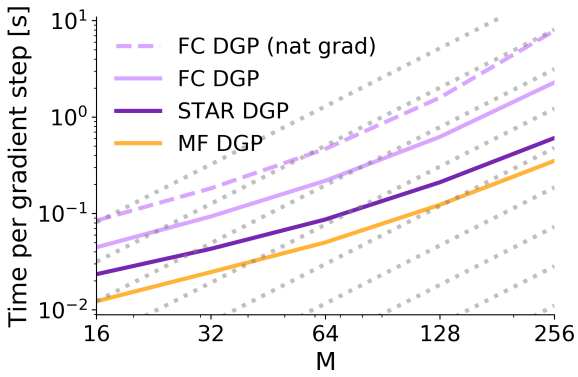


**Figure 3.6:** We compare the runtime of our efficient STAR DGP versus the FC DGP and the MF DGP on the *protein* UCI dataset. Shown is the runtime of one gradient step in seconds on a logarithmic scale as a function of the number of inducing points $M$. The dotted grey lines show the theoretical runtime $\mathcal{O}(M^2)$.

found to be the most important. Hence, our approach is (i) efficient as it allows to marginalise analytically over the global latent variables and (ii) expressive as it takes the most important couplings between the global latent variables across layers in the variational posterior into account. In a joint study of interpolation and extrapolation tasks as well as in a careful evaluation of the extrapolation task on its own, our approach outperforms its competitors (Salimbeni and Deisenroth, 2017; Havasi et al., 2018), since it balances accurate predictions and calibrated uncertainty estimates. Further research is required to understand why our structured approximations are especially helpful for the extrapolation task. One promising direction could be to look at differences of inner layer outputs (as done in Ustyuzhaninov et al., 2020b) and link them to the final deep GP outputs.

There has been a lot of follow-up work on deep GPs in which the probabilistic model is altered to allow for multiple outputs (Kaiser et al., 2018), multiple input sources (Hamelijnck et al., 2019), latent features (Salimbeni et al., 2019) or for interpreting the latent states as differential flows (Hegde et al., 2019). Our approach can be easily adapted to any of these models and is therefore a promising line of work to advance inference in deep GP models.

Our proposed structural approximation is only one way of coupling the latent GPs. Discovering new variational families that allow for more speed-ups either by applying Kronecker factorisations as done in the context of neural networks (Martens and Grosse, 2015), placing a grid structure over the inducing inputs (Izmailov et al., 2018), or by taking a conjugate gradient perspective on the objective (Wang et al., 2019) are interesting directions for future research. Furthermore, we think that the dependence of the optimal structural approximation on various factors (model architecture, data properties, etc.) is worthwhile to be studied in more detail.

CHAPTER 4

# Understanding Gaussian Process State-Space Models

Gaussian process state-space models (GPSSMs, see Sec. 2.4 for a detailed introduction) offer a promising possibility for probabilistic time series modelling (Wang et al., 2005; Frigola, 2015): By combining state-space models (see e.g. Särkkä, 2013) with Gaussian processes (GPs, see e.g. Rasmussen and Williams, 2006 or the introduction provided in Sec. 2.1) governing the transitions, GPSSMs are non-parametric, flexible, and probabilistic. However, inference in GPSSMs in inherently hard and analytically intractable: In order to model a time series $Y_T = \{y_t\}_{t=1}^T$, the state-space model introduces latent temporal states $X_{T_0} = \{x_t\}_{t=0}^T$ that separates the underlying dynamics from the measurements by allowing for observational noise, e.g. $p(y_t|x_t) = \mathcal{N}\left(y_t \middle| x_y, \sigma_y^2\right)$. The GP is used to model the transition function $f$ of these latent states, e.g. $p(x_t|x_{t-1}, f) = \mathcal{N}\left(x_t \middle| f(x_{t-1}), \sigma_x^2\right)$, where we introduced noise variances $\sigma_x^2$ and $\sigma_y^2$. The last step introduces another set of latent variables, the GP observations at the temporal latent states $F_T = \{f_t\}_{t=0}^{T-1}$ [where $f_t \equiv f(x_t)$] that need to be inferred along with the $X_{T_0}$ (see Sec. 2.4 for more details).

Early approaches did not scale to long time series (Wang et al., 2005; Frigola et al., 2013), which was remedied by relying on sparse GPs (Snelson and Ghahramani, 2005; Titsias, 2009; Hensman et al., 2013) and variational inference (Blei et al., 2017, see also Appx. A.2 for an introduction) in the approaches by Frigola et al. (2014) and Eleftheriadis et al. (2017). However, this leads to the introduction of yet another set of latent variables, the $M$ inducing outputs $F_M = \{f_m\}_{m=1}^M$ that, roughly speaking, summarise the information in the GP observations $F_T$ (where $M < T$).

The main difference between recent GPSSM approaches (Frigola et al., 2014; Eleftheriadis et al., 2017; Doerr et al., 2018; Ialongo et al., 2019) lies in the treatment of and the assumptions on the latent variables $\{X_{T_0}, F_T, F_M\}$ in the model. The earlier approaches assumed independence of the $X_{T_0}$ from the $\{F_T, F_M\}$ in the variational posterior which leads to the $\{F_T, F_M\}$ being analytically tractable for Frigola et al. (2014) (although only for specific kernels) and Eleftheriadis et al. (2017). Analytical tractability of latent variables is a desirable property as it is known to lead to better (and often faster) model convergence (Kingma et al., 2015) which we also saw in the previous chapter (see especially Fig. 3.2). However, in these cases the analytical tractability comes at the expense of model expressivity since the $X_{T_0}$ and the $\{F_T, F_M\}$ are assumed to be independent in the posterior. Furthermore, independence assumptions in variational inference are known to problematic since they can lead to an underestimation of predictive uncertainties (Turner and Sahani, 2011).

For these reasons, the most recent methods (Doerr et al., 2018; Ialongo et al., 2019) avoid the independence assumption but consequently need to find other ways to deal with the latent variables $F_T$ and $F_M$. Doerr et al. (2018) propose to use two approximations, first the fully-independent training conditional (FITC, see e.g. Sec. 2.2.1) approximation which makes the $F_T$ analytically tractable and then a biased approximation of an integral involving the $F_M$, which makes the latter also analytically tractable (see Sec. 2.4 for more details). In a follow-up work, Ialongo et al. (2019) criticise these two assumptions for their loss in accuracy and expressiveness and propose a method without the assumptions (amongst other differences) that relies on sampling both the $F_T$ and $F_M$ instead. This

constitutes a trade-off since sampling introduces additional variance and therefore often leads to worse convergence properties.

In this chapter, we theoretically study these important issues for general GPSSMs more closely: First, in Sec. 4.1, we study the role of the inducing outputs $F_M$ and seek a possibility to analytically marginalise them without assuming their independence from the $X_{T_0}$. While Doerr et al. (2018) used a biased approximation to treat the $F_M$ analytically and Ialongo et al. (2019) proposed using samples of the inducing outputs instead, we show that the $F_M$ can be marginalised exactly and analytically. We do this by building on the very similar integral that we solved in the previous chapter when dealing with the marginals of deep GPs (see Sec. 3.1) and similarly use a proof by induction. Second, we study the role of the FITC approximation in Sec. 4.2 which Doerr et al. (2018) used to achieve analytical tractability of the $F_T$. Following a similar analysis in Frigola et al. (2014), we find that the optimisation objective of Ialongo et al. (2019) stays the same whether the FITC approximation is used or not. Finally, in Sec. 4.3, we summarise our findings and discuss their implications, allowing us to motivate and map out the content of the following two chapters.

## 4.1 Analytical Marginalisation of the Inducing Outputs

Before we dive into the theoretical investigations of the current section we first briefly restate the problem at hand and also the proposed solutions of Doerr et al. (2018) and Ialongo et al. (2019): When using variational inference to treat GPSSMs, the optimisation objective (the ELBO) depends on the posterior marginals of the temporal latent states $q(x_t)$. This can be seen for the PRSSM method by Doerr et al. (2018) in Eq. (2.30) and for the VCDT method of Ialongo et al. (2019) in Eq. (2.36). For PRSSM, $q(x_t)$ is given as [cf. Eq. (2.31)]

$$q(x_t) = \int \left[ \int q(F_M) \prod_{t'=1}^{t} p(x_{t'}|x_{t'-1}, F_M) dF_M \right] q(x_0) dx_{0:t-1}, \tag{4.1}$$

where $p(x_{t'}|x_{t'-1}, F_M)$ is given in Eq. (2.28) and $q(x_0)$ and $q(F_M)$ are variational distributions that are typically Gaussians of the form $q(x_0) = \mathcal{N}(x_0|m_0, S_0)$ and $q(F_M) = \mathcal{N}(F_M|m_M, S_M)$, respectively, where the means and variances are variational parameters. Note that the $q(x_t)$ term for the VCDT approach in Eq. (A.30) is not exactly the same, but it has the same structure. Consequently, this leads to the same problem that we already had with Eq. (4.1), namely that neither the $F_M$ nor the $x_{0:t-1}$ can be easily integrated out.

As we already described in Sec. 2.4, Doerr et al. (2018) approximate the integral over the $F_M$ in Eq. (4.1) by exchanging the product and the integration for the inner integral which leads to the individual integrals inside the product being analytically tractable [cf. Eqs. (2.32) and (2.33)].[1] However, this approximation is biased and effectively removes all correlations between the GP outputs at different time points which is noted by Ialongo et al. (2019). They therefore propose to completely approximate the integration in Eq. (4.1) via sampling: In order to obtain a sample $x_t^{(s)}$ from $q(x_t)$, we first sample $F_M^{(s)} \sim q(F_M)$ and $x_0^{(s)} \sim q(x_0)$ and then recursively $x_{t'}^{(s)} \sim p(x_{t'}|x_{t'-1}^{(s)}, F_M^{(s)})$, which yields an unbiased estimate.

There are three reasons why we are not completely satisfied with this approach and that lead us to a more thorough investigation of $q(x_t)$: First, we saw in the previous chapter (more specifically in Fig. 3.2) that analytical marginalisation (as compared to sampling based marginalisation) is important for fast convergence, at least for the related problem of ELBO maximisation for deep GPs. Second, the integration in Eq. (4.1) very much resembles the integral in Eq. (3.1) which allows us to test whether

[1] The integration over the $x_{0:t-1}$ is still not analytically tractable which is then resolved by sampling.

the formalism that we developed to solve the latter can also be applied to the former. Therefore, we potentially obtain a better understanding of GPSSMs and their connection to deep GPs by reusing a previously established formalism. Third, we require a formula for $q(x_t)$ with the $F_M$ marginalised out in Chap. 5, which we discuss in more detail there.

In order to analytically marginalise the $F_M$ from $q(x_t)$, we exploit the similarity between the integrals in Eqs. (3.1) and (4.1): Comparing both, we see that the role of the latent GP observations $f_n^l$ for the deep GP is transferred to the temporal latent states $x_{t'}$ in the GPSSM setting and that consequently the role of the layers $l = 1, \ldots, L$ is assumed by the time points $t' = 1, \ldots, t$. Unsurprisingly, we can therefore use the same formalism, arriving at a very similar result [cf. Thm. 3.1] that we summarise in the following theorem (Longi et al., 2021).

**Theorem 4.1.** *The marginals of the latent state $q(x_t)$ in Eq. (4.1) at time point $t \in \{1, \ldots, T\}$, can be obtained as*

$$q(x_t) = \int q(x_0) \left[ \prod_{t'=1}^{t} q(x_{t'}|x_{t'-1}, \ldots, x_0) \right] dx_{0:t-1}, \tag{4.2}$$

*where all terms are Gaussian:*

$$q(x_t|x_{t-1}, \ldots, x_0) = \mathcal{N}\left(x_t \middle| \hat{\mu}_t, \hat{\Sigma}_t\right), \tag{4.3}$$

$$\hat{\mu}_t = \widetilde{\mu}_{t-1} + \widetilde{S}_{t-1,0:t-2} \widetilde{S}^{-1}_{0:t-2,0:t-2} \left(x_{1:t-1} - \widetilde{\mu}_{0:t-2}\right), \tag{4.4}$$

$$\hat{\Sigma}_t = \widetilde{S}_{t-1,t-1} - \widetilde{S}_{t-1,0:t-2} \widetilde{S}^{-1}_{0:t-2,0:t-2} \widetilde{S}_{0:t-2,t-1}. \tag{4.5}$$

*Here, the terms are given by*

$$\widetilde{\mu}_t = K_{tM} K_{MM}^{-1} m_M, \tag{4.6}$$

$$\widetilde{S}_{t,t'} = K_{tM} K_{MM}^{-1} S_M K_{MM}^{-1} K_{t'M}^\top + \delta_{tt'}(\sigma_x^2 + k_{tt} - K_{tM} K_{MM}^{-1} K_{t'M}^\top). \tag{4.7}$$

The notation $\cdot : \cdot$ is used to denote column vectors or submatrices, e.g. $x_{1:t} = \begin{pmatrix} x_1 & \cdots & x_t \end{pmatrix}^\top \in \mathbb{R}^t$, or $\widetilde{S}_{t-1,0:t-2} = \begin{pmatrix} \widetilde{S}_{t-1,0} & \cdots & \widetilde{S}_{t-1,t-2} \end{pmatrix} \in \mathbb{R}^{t-1}$. Furthermore, $\delta_{tt'}$ symbolises the Kronecker delta. Note that for $t = 1$ the slices in the additional terms of Eqs. (4.4) and (4.5) are empty and that therefore $\hat{\mu}_1 = \widetilde{\mu}_0$ and $\hat{\Sigma}_1 = \widetilde{S}_{0,0}$.

Similarly as for the proof of Thm. 3.1, we have a seemingly standard Gaussian integral in Eq. (4.1). However, the joint distribution of the latent states $x_1, ..., x_t$ cannot be seen as one joint multivariate Gaussian distribution since the latent state $x_{t'-1}$ enters the mean and the covariance of the Gaussian distribution of the following temporal latent state $x_{t'}$. Therefore, we again need to come up with a recurrent formulation of the problem that is amenable to a proof by induction which we sketch in the following. The full proof is provided in Appx. B.2.

**Sketch of the proof of Theorem 4.1**. We start the proof with Eq. (4.1), the general formula for the posterior marginals of the temporal latent states $q(x_t)$. In order to show the equivalence between the inner integral in Eq. (4.1) and the term in parentheses in Eq. (4.2) we proceed to find a new formula for the integrand in the inner integral of Eq. (4.1):

$$q(F_M) \prod_{t'=1}^{t} p(x_{t'}|x_{t'-1}, F_M) = q(F_M|x_t, \ldots, x_0) \prod_{t'=1}^{t} q(x_{t'}|x_{t'-1}, \ldots, x_0), \tag{4.8}$$

which holds for $t = 1, \ldots, T$. This is stated more formally in Lem. B.2 in Appx. B.2, in which we

also provide exact formulas for all terms. Importantly, all of them are multivariate Gaussians with known mean and covariance and we will show that the $q(x_{t'}|x_{t'-1},\dots,x_0)$ are as in Eqs. (4.3) – (4.5). The lemma itself can be proved by induction and we will show the general idea of the inductive step here: For this we assume that Eq. (4.8) holds for some $t$ (induction assumption) and we have to show that it then also holds for $t+1$.

Starting from the term of interest for $t+1$, we write

$$q(F_M)\prod_{t'=1}^{t+1}p(x_{t'}|x_{t'-1},F_M) = p(x_{t+1}|x_t,F_M)\left[q(F_M)\prod_{t'=1}^{t}p(x_{t'}|x_{t'-1},F_M)\right]$$
$$= p(x_{t+1}|x_t,F_M)q(F_M|x_t,\dots,x_0)\prod_{t'=1}^{t}q(x_{t'}|x_{t'-1},\dots,x_0), \quad (4.9)$$

where we used the induction assumption [Eq. (4.8)] in the last step. Comparing Eq. (4.9) with what we want to show, i.e., Eq. (4.8) for $t \to t+1$, we see that it remains to be shown that

$$p(x_{t+1}|x_t,F_M)q(F_M|x_t,\dots,x_0) = q(F_M|x_{t+1},\dots,x_0)q(x_{t+1}|x_t,\dots,x_0), \quad (4.10)$$

where $q(x_{t+1}|x_t,\dots,x_0)$ is as in Eqs. (4.3) – (4.5) (but with $t \to t+1$). In order to do so we start from the left hand side of Eq. (4.10) and build the joint distribution,

$$p(x_{t+1}|x_t,F_M)q(F_M|x_t,\dots,x_0) = q(F_M,x_{t+1}|x_t,\dots,x_0).$$

We then proceed and condition the resulting joint differently, i.e.,

$$q(F_M,x_{t+1}|x_t,\dots,x_0) = q(F_M|x_{t+1},\dots,x_0)q(x_{t+1}|x_t,\dots,x_0),$$

where it remains to be shown that these terms have the correct means and covariances. This derivation can be done using standard formulas for multivariate Gaussian distributions [Eqs. (B.3) – (B.6)].

Having proved Eq. (4.8) (or rather Lem. B.2), it becomes straightforward to prove Thm. 4.1: Starting from Eq. (4.1) and using Eq. (4.8), we obtain

$$q(x_t) = \int\left(\int q(F_M|x_t,\dots,x_0)dF_M\right)q(x_0)\left[\prod_{t'=1}^{t}q(x_{t'}|x_{t'-1},\dots,x_0)\right]dx_{0:t-1},$$

where we pulled all terms not depending on $F_M$ out of the inner integral. Since $q(F_M|x_t,\dots,x_0)$ is a properly normalised probability density, the inner integral equals one and only Eq. (4.2) remains. This completes the proof. □

As we discussed before Thm. 4.1, we had three reasons to try to analytically marginalise the $F_M$: i) faster convergence through replacing sampling with analytical formulas, ii) testing the applicability of the general proof formalism developed in order to prove Thm. 3.1 for deep GPs, and iii) obtaining analytical formulas in order to use them in another context to be discussed later. Clearly we have achieved iii) and this result will be very useful in Chap. 5 when we exploit a connection to stochastic differential equations (see e.g. Särkkä and Solin, 2019) to speed up inference for GPSSMs, especially for long time series. Furthermore, ii) has also been satisfactorily fulfilled since we could adapt the proof for Thm. 3.1 with ease to the proof of Thm. 4.1, even enabling us to reuse some of the steps as is discussed further in Appx. B.2. Only i) has not been fully successful: On the one hand, it is known (Kingma et al., 2015) and we have also empirically observed (see Fig. 3.2) that replacing sampling of a latent variable by its analytical marginalisation leads to faster convergence. On the other hand,

studying the analytical formulas in Eqs. (4.3) – (4.5) closely, we see that they involve an inversion of $\widetilde{S}_{0:t,0:t} \in \mathbb{R}^{t \times t}$. This implies a computational cost of $\mathcal{O}(t^3)$ for the analytical marginalisation whereas both approximations proposed in Doerr et al. (2018) and Ialongo et al. (2019) scale as $\mathcal{O}(t)$. Therefore, it is highly unlikely that using the analytical marginalisation translates to a faster algorithm in the case of GPSSMs. Note that for deep GPs the corresponding terms in Eqs. (3.4) and (3.5) also have a cubic dependence but with the number of layers in the deep GP instead of the number of time points in a time series. Whereas the number of layers in a deep GP is typically quite small (maybe 3 to 5), the number of time steps in a time series to be analysed by a GPSSM is roughly $\mathcal{O}(100 - 1000)$ (even if mini-batching is used). Hence the cubic scaling is not a problem for deep GPs while it renders the analytical marginalisation in GPSSMs impractical.

## 4.2 The Role of the FITC Approximation

Next, we turn our attention to the FITC approximation (Snelson and Ghahramani, 2005; Quinonero-Candela and Rasmussen, 2005) that Doerr et al. (2018) use in order to marginalise the $F_T$ and that Ialongo et al. (2019) criticise and abandon in order to arrive at a more expressive inference method. We already discussed the general idea as an early approach to sparse GPs around Eq. (2.10) and also the implications of using the FITC approximation in GPSSMs around Eq. (2.27). Due to the importance for the current section, we briefly recap the general idea here: Using the FITC approximation boils down to the assumption that the GP observations $F_T$ are independent given the inducing outputs $F_M$, i.e.,

$$p(F_T|X_T, F_M) \approx \prod_{t=0}^{T-1} p(f_t|x_t, F_M). \tag{4.11}$$

N.b. that this approximation preserves the correct marginals $p(f_t|x_t, F_M)$ and that therefore calculations that depend only on those will stay the same independent of whether FITC is used or not. The approximation works well in regions with densely spaced inducing inputs $X_M$ such that the prediction for $f_t$ at a new input point $x_t$ is narrowly confined by adjacent $F_M$ yielding a very narrow marginal $p(f_t|x_t, F_M)$. On the contrary, the approximation yields rather bad predictions $f_t$ and $f_{t'}$ for two points $x_t$ and $x_{t'}$ that are close to each other (in terms of the kernel length scales of the GP) but far away from all inducing inputs $X_M$. The marginals for $f_t$ and $f_{t'}$ would be quite broad in such a case and there would therefore be a high variance in either prediction using the FITC approximation. However, using the full $p(F_T|X_T, F_M)$ yields a high correlation between $f_t$ and $f_{t'}$ and having observed one would considerably narrow down the other. But if we keep this problematic behaviour in mind and use enough inducing points $X_M$ in the area where we expect the observations $x_t$ to be, we can still expect to obtain a good approximation by using Eq. (4.11).

The reason why we wish to study the role of the FITC approximation is threefold: First, we would expect that the method by Doerr et al. (2018) that uses the approximation performs worse than the method by Ialongo et al. (2019) that does not use it, but this is not what is reported (at least not consistently) in the latter work. Hence it could be worthwhile to investigate this discrepancy. Second, it would be interesting to see whether there is another option to marginalise the GP observations $F_T$ from the more expressive approach by Ialongo et al. (2019). Third, in Frigola et al. (2014, Appx. B.1) there is a remark about the FITC approximation stating that the optimisation objective of their GPSSM approach (which does not use the FITC approximation at all) stays the same if the FITC approximation were employed in both prior and approximate posterior. It is therefore worthwhile to study whether the same applies also for the model of Ialongo et al. (2019) in order to get a better understanding of that method and GPSSMs in general. We summarise our results in the following proposition (Lindinger et al., 2022):

**Proposition 4.2.** *Employing the FITC approximation [Eq. (4.11)] in the prior and approximate posterior [Eqs. (2.26) and (2.34), respectively] of the VCDT method of Ialongo et al. (2019) leaves the optimisation objective, the ELBO in Eq. (2.36), unchanged.*

This is the same result that Frigola et al. (2014) found for their method. The full proof can be found in Appx. B.3 and is conceptually very simple:

**Sketch of the proof of Proposition 4.2**. We start by plugging the FITC approximation [Eq. (4.11)] in Eqs. (2.26) and (2.34), the prior and approximate posterior of the VCDT method. Then, we plug those in Eq. (A.12), the general formula for the ELBO. Simplifying the resulting terms in the same way as we did in Appx. A.2.4 for the standard VCDT ELBO yields Eq. (2.36), i.e., the same result as without the FITC approximation. This completes the proof. □

A different phrasing of Prop. 4.2 could be: Regardless of whether the FITC approximation is used for training the VCDT method by Ialongo et al. (2019) or not, the final trained model would be the same.[2] This finding significantly weakens Ialongo et al.'s (2019) criticism of Doerr et al. (2018) for using the FITC approximation. It additionally opens up a possibility for marginalising the $F_T$ for the expressive GPSSM proposed by Ialongo et al. (2019) without losing any of the expressivity (during training). Hence, we will keep the FITC approximation in the following chapters.[3]

## 4.3 Chapter Summary, Conclusions and Outlook

In this chapter we set out to get a better understanding of modern GPSSMs by scrutinising and rethinking the different ways that have been proposed to deal with two sets of latent variables in the models, the GP observations $F_T$ and the inducing outputs $F_M$. The latter play an important role for obtaining the posterior marginals of the temporal latent states, $q(x_t)$, and we compared the proposed approaches in Sec. 4.1. Moreover, we found that while an analytical marginalisation of the $F_M$ in $q(x_t)$ is possible, the resulting computations scale as $\mathcal{O}(t^3)$ which in itself is impractical for efficient inference in GPSSMs. Then, in Sec. 4.2, we analysed the role of the FITC approximation, which Doerr et al. (2018) propose to be able to analytically deal with the $F_T$ and which is criticised by Ialongo et al. (2019). Here, we found that the FITC approximation plays a much smaller role than it seems from the criticism of Ialongo et al. (2019). Indeed, we showed that the optimisation objective of their VCDT method does not change whether the FITC approximation is used or not.

From these findings we directly draw the following conclusions: First, while the formulas for the analytical marginalisation of the $F_M$ from $q(x_t)$ in Thm. 4.1 reveal an interesting connection to a similar result for deep GPs (cf. Thm. 3.1), their direct application to inference in GPSSMs is impractical due to the $\mathcal{O}(t^3)$ scaling. Therefore, we agree in this point with Ialongo et al. (2019) and recommend the usage of their sampling based approach to approximate $q(x_t)$ which yields an unbiased estimate in contrast to the approximation by Doerr et al. (2018) while keeping the same $\mathcal{O}(t)$ scaling. We will use this approach in Chap. 5. Note however, that in the experiments of Ialongo et al. (2019) it can be seen that the approach by Doerr et al. (2018) performs almost identically whether or not the unbiased approximation via sampling is used. Second, we deem avoiding the FITC approximation for the training of GPSSMs unnecessary since all variational inference methods either use it directly (Doerr et al., 2018) or obtain the same optimisation objective with or without it

[2] Note that there still remains a difference during prediction where the FITC approximation changes the outcome. However, this could also be changed for the method by Doerr et al. (2018), i.e., train with the FITC approximation and predict without it.

[3] Note that the PRSSM approach by Doerr et al. (2018) would not efficiently be possible without the FITC approximation as this inevitably leads to a $\mathcal{O}(t^3)$ scaling for obtaining $q(x_t)$. This can be seen by deriving the ELBO as in Appx. A.2.3 but without using the FITC approximation.

(Frigola et al., 2014; Eleftheriadis et al., 2017; Ialongo et al., 2019).[4] Consequently, we will rely on the FITC approximation in Chaps. 5 and 6.

However, while these conclusions provide a better understanding of GPSSMs they do not directly present obvious ways in which to significantly improve upon the state-of-the-art inference methods of Doerr et al. (2018) and Ialongo et al. (2019). Note that a small improvement could potentially be gained by by using the FITC approximation with the method of Ialongo et al. (2019). As we saw, this would leave the optimisation objective unchanged, but it would allow marginalising the $F_T$ from the model (in the same way as in Appx. A.2.3) which removes the need to sample the $F_T$ which in turn should reduce the variance during training. However, we shall be interested in more than such incremental changes for the remainder of this thesis and by building on the observations of the current chapter we will be able to obtain significant improvements as we discuss below:

One of the reasons to attempt the analytical marginalisation in Sec. 4.1 is derived from our findings for the deep GP in Chap. 3. There, we saw that analytical marginalisation leads to faster convergence as opposed to sampling based marginalisation (see especially Fig. 3.2). Our hope was that this result could directly be transferred to inference in GPSSMs. While this was not the case, we will still be able to use the results of Sec. 4.1 to make inference in GPSSMs more efficient although by considering a different aspect. In the following chapter [Chap. 5], we will consider the capability of GPSSM approaches to deal with long time series (or rather mini-batches) that are necessary when there is a slowly varying trend in the data that only manifests itself over hundreds or thousands of time points. Using a connection to stochastic differential equations (Särkkä and Solin, 2019) we will then propose a method that exploits the theoretical result in Thm. 4.1 and can efficiently deal with such time series. This new approach generally improves the efficiency of GPSSMs applied to long time series and also improves the prediction for time series data with slow and fast effects (towards which it is targeted).

However, this proposed approach does not generally improve over the other state-of-the-art methods. In the current chapter we only studied the differences between GPSSMs in the treatment of the latent variables $F_T$ and $F_M$ but ignored the role of the temporal latent states $X_{T_0}$. Having assessed that the differences in the treatment of the $F_T$ and $F_M$ play only minor roles for the performance of recent GPSSM approaches, the obvious choice for further scrutiny if we wish to improve the latter are the temporal latent states. In Chap. 6 we will therefore discuss the differences in the treatment of the $X_{T_0}$ between the modern approaches that take conditional dependencies between all latent variables into account (Doerr et al., 2018; Ialongo et al., 2019). We will find that the new form of posterior over the $X_{T_0}$ that Ialongo et al. (2019) propose actually constitutes a trade-off: While the posterior is clearly more expressive than the one used by Doerr et al. (2018), it also introduces many additional parameters that have to be learned thus leading to a harder inference problem. (We will find that the additional parameters are especially problematic since they have to be learned sequentially, leading to a difficult optimisation problem.) Exploring this trade-off further, we will come up with a new inference method that is more expressive than the one by Doerr et al. (2018) but has no additional parameters and thus leads to an easier learning problem than the method by Ialongo et al. (2019).

[4] For the method by Eleftheriadis et al. (2017) the same reasoning as in Frigola et al. (2014, Appx. B.1) applies since both methods use the same variational approximation.

CHAPTER 5

# Multi-Resolution Gaussian Process State-Space Models

In many different fields, e.g. epidemiology (Zimmer and Yaesoubi, 2020), finance (Heaton et al., 2017), or engineering (Yu et al., 2020), time series modelling is a core task (see also Sec. 1.2.3). However, often the underlying physical model is unknown and the dynamics have to be learned directly from data. In order to support arbitrary dynamics, a time series model should ideally be non-parametric, and since characteristic features (e.g. rapid transitions in the dynamics) are often rare events, the time series model should take this into account by relying on probabilistic techniques. Gaussian process state-space models (GPSSMs, see Sec. 2.4 for technical details) fulfil both these requirements by modelling non-linear, unknown dynamics via a state-space model using a Gaussian process (GP) prior in the transition function (Wang et al., 2005; Frigola, 2015).

Early GPSSM approaches (Wang et al., 2005; Frigola et al., 2013) scaled poorly to long time series, which was partially remedied by relying on variational inference and sparse Gaussian processes (see Sec. A.2 for an introduction) as an approximate inference method (e.g. Frigola et al., 2014; Eleftheriadis et al., 2017). Some of the approaches (Frigola et al., 2014; Ialongo et al., 2019) are not designed to deal with mini-batches (or rather subsequences of the time series) and would therefore encounter problems with large runtimes and memory footprints as well as the problem of vanishing and exploding gradients (Pascanu et al., 2013) when being applied to long trajectories. Note that existing methods that solve these problems for recurrent neural networks (Hochreiter and Schmidhuber, 1996; Chung et al., 2014) are not directly applicable to GPSSM models. While the recent methods of Eleftheriadis et al. (2017) and Doerr et al. (2018) can use subsequences, a general problem when using mini-batches for time series modelling persists: Slow effects that manifest only over a time spanned by multiple subsequence lengths cannot be inferred (Williams and Zipser, 1995).

The goal of this chapter is therefore to provide a solution to this problem for GPSSMs and we will do so by proposing an extension to the method by Doerr et al. (2018): We introduce an approach that works with multiple components, each of which has its own resolution. A component with a high resolution is well suited to describing short-term effects in the data by taking densely spaced observations of the time series into account. In contrast, long-term effects are better captured by low-resolutions components that take diluted observations of the time series, e.g., only every tenth or hundredth observation, into account. This reasoning is sketched in Fig. 5.1.

As we will see below, naively training such a model is inefficient which we overcome by employing the backfitting algorithm (Breiman and Friedman, 1985) to learn the different components and use a novel technique to sample from the model. The latter builds on connecting our theoretical results from Chap. 4 (especially Thm. 4.1) to stochastic differential equations (see e.g. Särkkä and Solin, 2019).

The structure of this chapter is as follows: First, in Sec. 5.1 we give a primer on stochastic differential equations and briefly review some of the work that also uses them in connection with GPs or GPSSMs. Then we present our proposed model and show how to efficiently train it by introducing the points mentioned above (Sec. 5.2). In Sec. 5.3 we show that our approach compares favourably to state-of-the-art methods on two semi-synthetic data sets and on a complicated engine modelling task.

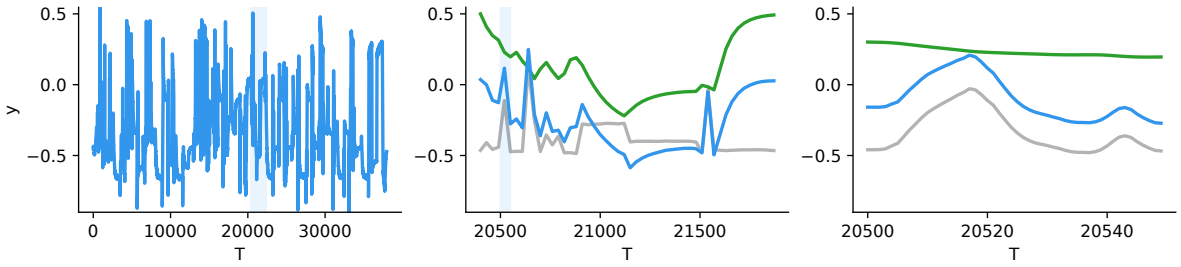


**Figure 5.1:** Depicted is a semi-synthetic one-dimensional time series dataset (blue) that is created as the sum of a slowly varying function (green) and a fast varying one (grey). On the left we show the complete trajectory. In the middle we zoom in on the shaded blue area marked in the left plot (note the different x-axis) and show the groundtruth diluted by a factor of 30, i.e., the blue curve shows only every 30-th observation. This allows a good representation of the slowly varying function (green). On the right we have zoomed in once again, here on the shaded blue area marked in the centre plot. For this subsequence every observation of the time series in blue is shown. These densely spaced observations give a faithful representation of the fast varying function (grey). Reproduced from Longi et al. (2022).

Finally, in Sec. 5.4, we summarise the current chapter.

## 5.1 Stochastic Differential Equations and Gaussian Processes

Stochastic differential equations (SDEs, see e.g. Särkkä and Solin, 2019) are generalisations of ordinary differential equations (ODEs), where stochasticity is introduced via a driving random process (sometimes called stochastic process). Using Brownian motion $W_t$ as the driving process, SDEs can be written as

$$dx(t) = g(x(t), t)dt + L(x(t), t)dW_t, \tag{5.1}$$

where we use the differential notation $d\cdot$ to denote infinitesimal changes in the corresponding variables. Here $x(t)$ is the time-dependent variable (or state of a system) that we are interested in, $g$ is the so-called *drift* term that describes the deterministic change of the variable, $L$ the so-called *diffusion* (sometimes also dispersion) term that describes the randomness through a coupling to the stochastic process $W_t$. Finally, Brownian motion (sometimes also called Wiener process) is defined by three properties: i) it starts at the origin, $W_0 = 0$, ii) increments of the Brownian motion follow a zero-mean Gaussian distribution,

$$\Delta W_t = W_t - W_{t'} \sim \mathcal{N}(0, \Delta t), \qquad \Delta t = t - t', \tag{5.2}$$

and iii) increments are independent if the time spans do not overlap.[1] Solving the SDE in Eq. (5.1) typically amounts to being given an initial value $x_0$ [or often also a distribution $p(x_0)$] at the initial time $t_0$ and then finding the distribution over the variable $x(t)$ at an arbitrary time point $t > t_0$. Since this is already a very hard problem for ODEs, i.e., Eq. (5.1) without the stochastic component, where typically numerical solutions are required, it is not surprising that this becomes even harder for SDEs. Matters are made worse by the fact that even for simple SDEs the naive approach of trying to simply integrate both sides of Eq. (5.1) from $t_0$ to $t$ (with the typical Riemannian integral) fails due to the stochasticity introduced by $W_t$. Fortunately there is a solution to this, namely Itô calculus which boils down to introducing a new integral formulation that fixes the problems associated with the stochasticity of the Brownian motion (see e.g. Särkkä and Solin, 2019, Sec. 4.1 details and derivations).

[1] Generally the Brownian motion has an additional diffusion coefficient $Q$ that changes Eq. (5.2) to $\Delta W_t \sim \mathcal{N}(\Delta W_t|0, Q\Delta t)$. In this work we set $Q = 1$.

However, most practical problems do not allow for analytical solutions of the resulting integrals such that we have to use numerical approximations. The simplest approach for SDEs is a direct generalisation of Euler's method for ODEs (see e.g. Särkkä and Solin, 2019, Sec. 2.6) leading to the so-called Euler-Maruyama method: In order to approximately propagate the state $x(t_0)$ forward in time according to the SDE in Eq. (5.1) from time $t_0$ to $t_1$, we linearise the dynamics and approximate

$$x(t_1) \approx x(t_0) + g(x(t_0), t_0)\Delta t_1 + L(x(t_0), t_0)\Delta W_{t_1}, \tag{5.3}$$

where $\Delta t_1 = t_1 - t_0$ and $\Delta W_{t_1}$ is as in Eq. (5.2). Note that Eq. (5.3) can be equivalently written as

$$x(t_1) \approx x(t_0) + \Delta x_{t_0}, \qquad \Delta x_{t_0} \sim \mathcal{N}\left(g(x(t_0), t_0)\Delta t_1, L^2(x(t_0), t_0)\Delta t_1\right), \tag{5.4}$$

where we used the fact that $x \sim \mathcal{N}\left(\mu, \sigma^2\right)$ is equivalent to $x = \mu + \sigma\epsilon$ with $\epsilon \sim \mathcal{N}(0, 1)$ (a so-called reparameterisation). The approximation in Eq. (5.3) becomes better the smaller $\Delta t_1$ and can be used recursively to simulate $x(t_1),\ x(t_2), \ldots,\ x(t_T)$ until time $t_T$. For further general details about SDEs, more accurate numerical solution methods or practical examples we refer the interested reader to Särkkä and Solin (2019).

Finally, since we are planning to use SDEs in connection with GPs and GPSSMs below, we briefly review some related work in the following. In a method called differential Gaussian process flows, Hegde et al. (2019) introduce an SDE of the form

$$dx_t = \mu(x_t)dt + \sqrt{\Sigma(x_t)}dW_t,$$

where $\mu$ and $\Sigma$ are taken to be the mean and covariance of a sparse Gaussian process [see Eq. (2.7)]. Applying the Euler-Maruyama method here leads to increments $\Delta x_t \sim \mathcal{N}(\mu(x_t)\Delta t, \Sigma(x_t)\Delta t)$ [cf. Eq. (5.4)] which is essentially the same as the sampling step through the layers of doubly stochastic deep GPs (Salimbeni and Deisenroth, 2017) in Eq. (2.23). The method of Hegde et al. (2019) therefore provides a new view on deep GPs through the lens of SDEs in which a few different layers are replaced by applying the same layer of GPs many times (which leads to less parameters that have to be inferred).

Other related methods (e.g. Ruttor et al., 2013; Yıldız et al., 2018; Zhao et al., 2020) are aiming at learning the drift term of general SDEs from data by placing a GP prior on it and inferring the posterior of the latter. This approach corresponds to the SDE

$$dx_t = f(x_t)dt + \sigma_\Delta dW_t, \tag{5.5}$$

where we place a GP prior on $f$ and only assume a simple constant diffusion $\sigma_\Delta$ since the GP is already stochastic and can learn arbitrary mappings (Yıldız et al., 2018, additionally place a GP prior on the diffusion and learn its posterior as well). Applying the Euler-Maruyama method to Eq. (5.5) yields

$$\Delta x_t \sim \mathcal{N}\left(f(x_t)\Delta t, \sigma_\Delta^2 \Delta t\right) \tag{5.6}$$

for the increment $\Delta x_t$ [cf. Eq. (5.4)]. Using $x_{t+1} = x_t + \Delta x_t$, it is easy to see that Eq. (5.6) therefore corresponds to a transition model of the form

$$p(x_{t+1}|f_t, x_t) = \mathcal{N}\left(x_{t+1}\middle| x_t + f_t\Delta t, \sigma_\Delta^2 \Delta t\right)$$

Comparing this to the GPSSMs that we introduced in Sec. 2.4, we can see that Eq. (5.6) corresponds

essentially to the transition model for GPSSMs [Eq. (2.25)], which we repeat here for convenience:

$$p(x_t|f_{t-1}, x_{t-1}) = \mathcal{N}\left(x_t \middle| f_{t-1}, \sigma_x^2\right).$$

The only differences are the additional $\Delta t$ which will be the focus of a big part of Sec. 5.2 and the $x_t$ in the mean, which we will also discuss there. In a recent work, Zhao et al. (2020) improved upon this approach by considering a higher order approximation instead of the Euler-Maruyama scheme in Eq. (5.6) and by additionally employing a state-space GP for $f$ (see Särkkä et al., 2013 or the brief introduction in Sec. 2.2.3).

## 5.2 Training Gaussian Process State-Space Models on Multiple Resolutions

Having established the theoretical foundations on SDEs in the previous section and on GPSSMs in Sec. 2.4, we are ready to introduce the multi-resolution GPSSM and explain how it is trained in this section. We introduce the new model formulation in Sec. 5.2.1 and then show how it can be efficiently trained in Sec. 5.2.2.

### 5.2.1 Model Formulation

So far we have only dealt with a single component in the model formulation of GPSSMs [Eqs. (2.25) and (2.27)] and for the approximate posterior [Eq. (2.29)]. In the following we extend this formulation to multiple components where we build on the approach by Doerr et al. (2018) since it naturally allows for using mini-batches.

We separate the latent states $x_t \in \mathbb{R}^{d_x}$ into $C$ components such that $x_t = \{x_t^{(c)}\}_{c=1}^C$, with $x_t^{(c)} \in \mathbb{R}^{d_c}$ and $\sum_{c=1}^C d_c = d_x$. Assuming that the different components evolve independently from each other leads to the augmented prior model [cf. Eq. (2.27)]

$$p(Y_T, X_{T_0}, F_M) = \prod_{c=1}^{C} \left[p(x_0^{(c)})p(F_M^{(c)})\right] \prod_{t=1}^{T} \left[p(y_t|x_t) \prod_{c=1}^{C} p(x_t^{(c)}|x_{t-1}^{(c)}, F_M^{(c)})\right]. \tag{5.7}$$

Here $p(x_0^{(c)})$ is as in Eq. (2.25) for each component and $p(F_M^{(c)}) = \mathcal{N}\left(F_M^{(c)} \middle| 0, K_{MM}^{(c)}\right)$ is the GP prior on the inducing outputs per component with $K_{MM}^{(c)} = \{k^{(c)}(x_m^{(c)}, x_{m'}^{(c)})\}_{m,m'=1}^M$ with generally different kernels $k^{(c)}$ and inducing inputs $X_M^{(c)} = \{x_m^{(c)}\}_{m=1}^M$ per component. We keep the emission model $p(y_t|x_t)$ from Eq. (2.25) that is the only part in our model that connects the different components. The transition model is as in Eq. (2.28) with the important difference that inputs to the kernels are not given by the complete latent state but only by the corresponding component such that e.g. $K_{t-1,M}^{(c)} = \{k^{(c)}(x_{t-1}^{(c)}, x_m^{(c)})\}_{m=1}^M$.[2]

Ignoring for a moment that we wanted to use different resolutions for the different components, we can perform inference in the model in Eq. (5.7) by generalising the variational inference approach of Doerr et al. (2018) which leads to the approximate posterior [cf. Eq. (2.29)],

$$q(X_{T_0}, F_M) = \prod_{c=1}^{C} \left[q(x_0^{(c)})q(F_M^{(c)}) \prod_{t=1}^{T} p(x_t^{(c)}|x_{t-1}^{(c)}, F_M^{(c)})\right]. \tag{5.8}$$

[2] This is different from the way in which multi-dimensional temporal latent states are typically treated in GPSSMs, see e.g. the detailed explanation in the paragraph about multi-dimensional latent states in Lindinger et al. (2020, Appx. E.2).

Here, we choose Gaussian variational families per component, $q(x_0^{(c)}) = \mathcal{N}\left(x_0^{(c)} \middle| m_0^{(c)}, S_0^{(c)}\right)$ and $q(F_M^{(c)}) = \mathcal{N}\left(F_M^{(c)} \middle| m_M^{(c)}, S_M^{(c)}\right)$ with the free variational parameters $\psi^{(c)} = \{m_0^{(c)}, S_0^{(c)}, m_M^{(c)}, S_M^{(c)}\}$. In addition to inferring these, we also have to optimise the parameters of the emission model $p(y_t|x_t$ [see Eq. (2.25)] and the per-component model parameters $\theta^{(c)}$ which are given by the hyperparameters of the kernel $k^{(c)}$ associated with each component and the transition noise standard deviation $\sigma_x^{(c)}$ [see Eq. (2.28)]. As usual in variational inference, we do so by optimising the ELBO [Eq. (A.12)], which for our multi-resolution (MR) approach (following a very similar derivation as in Appx. A.2.3), can be shown to equal

$$\mathcal{L}_{\text{MR}} = \sum_{t=1}^{T} \mathbb{E}_{q(x_t)} \left[\log p(y_t|x_t)\right] - \sum_{c=1}^{C} \left(\text{KL}\left[q(x_0^{(c)}) \parallel p(x_0^{(c)})\right] + \text{KL}\left[q(F_M^{(c)}) \parallel p(F_M^{(c)})\right]\right). \quad (5.9)$$

Here, the KL-divergences are analytically tractable and the marginals of the temporal latent states are given by $q(x_t) = \prod_{c=1}^{C} q(x_t^{(c)})$, where each of the terms $q(x_t^{(c)})$ is given by the equivalent of Eq. (2.31). As we discussed in detail in Sec. 4.1, these terms are analytically intractable but there exist several approximation approaches. In this work we adopt the sampling scheme proposed by Ialongo et al. (2019).

So far, this has been a relatively straightforward extension of the GPSSM model by Doerr et al. (2018) with the only difference that we have introduced a structured latent space with $C$ components. This will change in the following when we consider how this model can be most efficiently applied to long time series with fast and slow dynamics as depicted in Fig. 5.1. For this we require i) a way to efficiently update the parameters of the different components, ii) a possibility to generally deal with long time series without running into numerical issues, and finally iii) a way to efficiently sample from $q(x_t^{(c)})$, especially if we are using a component with a low resolution.

For i) we exploit that we assumed independence of the $C$ components in the prior and in the approximate posterior in Eqs. (5.7) and (5.8), respectively which led also to the independence of the marginals $q(x_t)$ which are required for computing our optimisation objective in Eq. (5.9). We can therefore employ an iterative method and decide to use the backfitting algorithm (Breiman and Friedman, 1985): In this approach, we cycle through the $C$ components, fixing the parameters of all but the $c$-th component to find the optimal parameters $\theta^{(c)}$ and $\psi^{(c)}$ and then continue with the next component. While this is already more efficient than updating all parameters simultaneously, we can additionally reuse the samples from the $q(x_t^{(c)})$ multiple times since this distribution does not change unless the $c$-th component is updated in the current cycling step. Especially this second part is important later when we want to use components with different resolutions.

Continuing with ii), we make use of mini-batches (see e.g. Bottou, 2010) or rather subsequences which are well established also for time series (e.g. Aicher et al., 2019) and also used by Doerr et al. (2018). This is enabled by the ELBO in Eq. (5.9) decomposing between the data points such that we can estimate

$$\sum_{t=1}^{T} \mathbb{E}_{q(x_t)} \left[\log p(y_t|x_t)\right] \approx \frac{T}{B} \sum_{t=t_0}^{t_0+B} \mathbb{E}_{q(x_t)} \left[\log p(y_t|x_t)\right], \quad (5.10)$$

where $B$ is the batch-size and $t_0$ is the initial time index of the batch under consideration. In order to efficiently sample from $q(x_t)$ without having to start from $x_0$ for every batch we follow Aicher et al. (2019) and break the temporal dependency between $x_0$ and $x_{t_0}$ by introducing a buffer of size $B_0$. We assume that this buffer is sufficient to obtain (nearly) unbiased samples from $q(x_{t_0})$ when we sample $q(x_{t_0-B_0}) \sim q(x_0)$ and continue from there according to the transition model. This requires much less samples than having to start from a sample of the initial latent state distribution $x_0$ and then recursively sampling $x_t$ for $t = 1, \ldots, t_0$. Note that while introducing this buffer helps reducing

the bias of the approximation in Eq. (5.10), this approach is still not completely unbiased. This is especially the case if there are effects that evolve slower than the length of one mini-batch $B$ and that can consequently not be captured by the approximation.

This indirectly leads us to iii) and the reason to use different resolutions: If we consider a component with a low resolution that takes only every $R$-th observation into account (with $R > 1$), we can pack a longer history into a mini-batch of the same size (compared to using the standard resolution that corresponds to $R = 1$). This can further resolve the issues with the biased approximation in Eq. (5.10). However, if we consider only components with a low resolution we will not be able to resolve fast varying dynamics (see Fig. 5.1, right). Therefore, we plan to use different resolutions for the different components, i.e., the $c$-th component takes only every $R^{(c)}$-th observation into account. Exemplarily using a batch size $B$, a resolution $R^{(c)}$, and starting at the time index $t_0$ this corresponds to using time indices

$$\mathcal{T} = \{t_0 + bR^{(c)}\}_{b=0}^{B}, \tag{5.11}$$

which can be used instead of the consecutive indices in Eq. (5.10). Unfortunately, with this approach we undermine the solution to the problem ii), since having a component with a low resolution (corresponding to a large $R^{(c)}$) means we have to sample from $q(x_{t_0 - R^{(c)}B_0})$ to $q(x_{t_0+R^{(c)}B})$ if we want to keep the same batch size $B$. This amounts to an increase of a factor $R^{(c)}$ in sampling time which is impractical especially for large $R^{(c)}$ and might additionally lead to numerical issues with the gradients due to the long subsequences required for such an approach. In order to overcome this new issue, we draw on the connection between the discretized GP SDE in Eq. (5.6) and the transition model of the GPSSM in Eq. (2.28): We show in the next section in detail that there exists a parameterisation of the discretised GP SDE (interpreted as a GPSSM) such that this model is equivalent to a canonical GPSSM. Using this connection we propose a theoretically grounded way to draw mini-batches of approximate samples from $q(x_t)$ for different resolutions with a constant time requirement.

### 5.2.2 Equivalence Between Discretised GP SDEs and GPSSMs

The goal of this section is to solve the final efficiency problem of the proposed multi-resolution GPSSM approach of the previous section, namely efficiently drawing temporal latent state samples for components with a low resolution (which require a long history). In order to do so we will sparsify the discretised GP SDE transition in Eq. (5.6) and embed it in a state-space model, yielding an SDE GPSSM. We will then proceed to show that for a certain parameterisation of the latter model we can get the same temporal latent state samples and consequently also the same optimisation objective as for a canonical GPSSM if we use the same resolution. This will then allow us to use the models interchangeably and we will use the SDE GPSSM model to efficiently draw approximate samples from $q(x_t)$ for low resolutions.

We start by embedding Eq. (5.6) in a GPSSM. For this we use a different indexing than for standard GPSSMs in Sec. 2.4 in order to clearly distinguish the two approaches: We used $x_t$ for canonical GPSSMs to denote the latent state at time $\tau = t\Delta_t$ (starting at $\tau = 0$), where $\Delta_t$ is the (constant) data set dependent time difference between two subsequent observations in the time series. For SDE GPSSMs we use $x_j$ instead to denote a latent state at time $\tau = jR\Delta_t$, where $R$ controls the resolution and there are $J = T/R$ latent states in the SDE GPSSM. For the latter, we additionally decorate every distribution or parameter with a $\Delta$ in order to clearly distinguish it from the canonical GPSSM. Interpreting the discretised SDE increments in Eq. (5.6) as a GPSSM transition model and using the described notation, we obtain

$$p^{\Delta}(x_j | f_{j-1}, x_{j-1}) = \mathcal{N}\left(x_j \middle| x_{j-1} + f_{j-1} R\Delta_t, R\Delta_t \sigma_{\Delta}^2\right), \tag{5.12}$$

where we introduced $f_j \equiv f(x_j)$ and replaced $\Delta t$ by $R\Delta_t$, where at the moment $R$ is still arbitrary and can be chosen such that Eq. (5.12) matches Eq. (5.6).[3] Next, we proceed to sparsify the GP (see Sec. 2.2.1) and, following Doerr et al. (2018) use the FITC approximation (see the discussion in Sec. 4.2). This means we assume

$$p^\Delta(F_J, F_M | X_{J_0}) \approx p^\Delta(F_M) \prod_{j=0}^{J-1} p^\Delta(f_j | x_j, F_M), \tag{5.13}$$

where $F_J \equiv \{f_j\}_{j=0}^{J-1}$, $X_{J_0} \equiv \{x_j\}_{j=0}^{J}$ and $p^\Delta(f_j|x_j, F_M)$ is as in Eq. (A.24) (with $t$ replaced by $j$). Finally, choosing an initial latent state distribution $p^\Delta(x_0)$ and an emission model $p^\Delta(y_j|x_j)$ (which we currently leave unspecified), we have our SDE GPSSM prior model,

$$p^\Delta(Y_J, X_{J_0}, F_J, F_M) = p^\Delta(x_0) p^\Delta(F_J, F_M | X_{J_0}) \prod_{j=1}^{J} p^\Delta(y_j|x_j) p^\Delta(x_j | f_{j-1}, x_{j-1}).$$

By marginalising the $F_J$ from this model [exactly as in Appx. A.2.3], we arrive at

$$p^\Delta(Y_J, X_{J_0}, F_M) = p^\Delta(x_0) p^\Delta(F_M) \prod_{j=1}^{J} p^\Delta(y_j|x_j) p^\Delta(x_j | x_{j-1}, F_M), \tag{5.14}$$

$$p^\Delta(x_j | x_{j-1}, F_M) = \mathcal{N}\left(x_j \middle| x_{j-1} + R\Delta_t \mu^\Delta(x_{j-1}, F_M), R\Delta_t \sigma_\Delta^2 + (R\Delta_t)^2 \Sigma^\Delta(x_{j-1})\right), \tag{5.15}$$

$$\mu^\Delta(x_{j-1}, F_M) = K^\Delta_{j-1,M} \left(K^\Delta_{MM}\right)^{-1} F_M, \tag{5.16}$$

$$\Sigma^\Delta(x_{j-1}) = k^\Delta_{j-1,j-1} - K^\Delta_{j-1,M} \left(K^\Delta_{MM}\right)^{-1} \left(K^\Delta_{j-1,M}\right)^\top. \tag{5.17}$$

Here, e.g. $K^\Delta_{j-1,M} = \{k^\Delta(x_{j-1}, x^\Delta_m)\}_{m=1}^M$, where $k^\Delta$ is the kernel of the GP in the SDE and $X^\Delta_M = \{x^\Delta_m\}_{m=1}^M$ is the set of inducing inputs used for this GP.

At this point we can already compare the model in Eqs. (5.14) – (5.17) to the canonical model in Eqs. (2.27) and (2.28), where we immediately see two differences. One is the (logical) appearance of $R\Delta_t$ terms, the other that there is an additional $x_{j-1}$ in the mean of Eq. (5.15) compared to Eq. (2.28). The latter difference is also easily explained since GPSSMs describe *transitions* $x_t \to x_{t+1}$, while (discretised) SDEs describe *differences* $x_j \to x_j + \Delta j$. However, we can also consider canonical GPSSMs with a residual transition model $p(x_{t+1}|x_t, f_t) = \mathcal{N}\left(x_{t+1} \middle| x_t + f_t, \sigma_x^2\right)$ which leads to [cf. Eq. (2.28)]

$$p(x_t | x_{t-1}, F_M) = \mathcal{N}\left(x_t \middle| x_{t-1} + K_{t-1,M} K_{MM}^{-1} F_M, \sigma_x^2 + k_{t-1,t-1} - K_{t-1,M} K_{MM}^{-1} K_{t-1,M}^\top\right). \tag{5.18}$$

Note that the residual transition model is equivalent (or simply a different parameterisation) to using a GP with a linear mean function which Doerr et al. (2018) and Ialongo et al. (2019) use in their experiments.

After this small aside we turn our attention back to the model in Eqs. (5.14) – (5.17): Inference is equally intractable as for the model in Eqs. (2.27) and (2.28) and we therefore choose the same

[3] Note that Eq. (5.6) described increments $\Delta x_t = x_t - x_{t-1}$, whereas Eq. (5.12) gives a distribution over $x_j$. Adjusting the indices and using $x_j = x_{j-1} + \Delta x_j$, it is easy to see that Eqs. (5.6) and (5.12) are equivalent for $R = 1$.

approach, variational inference with an equivalent variational family [cf. Eq. (2.29)]

$$q^{\Delta}(X_{T_0}, F_M) = q^{\Delta}(x_0) q^{\Delta}(F_M) \prod_{j=1}^{J} p^{\Delta}(x_j | x_{j-1}, F_M). \tag{5.19}$$

Here $q^{\Delta}(x_0) = \mathcal{N}\left(x_0 \middle| m_0^{\Delta}, S_0^{\Delta}\right)$, $q^{\Delta}(F_M) = \mathcal{N}\left(F_M \middle| m_M^{\Delta}, S_M^{\Delta}\right)$, and the $p^{\Delta}$ are given in Eq. (5.15). Consequently, this leads to an equivalent ELBO to the one for the canonical GPSSM formulation [see Eqs. (2.30) and (2.31)]:

$$\mathcal{L}_{\text{SDE}} = \sum_{j=1}^{J} \mathbb{E}_{q^{\Delta}(x_j)} \left[\log p^{\Delta}(y_j | x_j)\right] - \text{KL}\left[q^{\Delta}(x_0) \parallel p^{\Delta}(x_0)\right] - \text{KL}\left[q^{\Delta}(F_M) \parallel p^{\Delta}(F_M)\right], \tag{5.20}$$

$$q^{\Delta}(x_j) = \int q^{\Delta}(x_0) \left[\int q^{\Delta}(F_M) \prod_{j'=1}^{j} p^{\Delta}(x_{j'} | x_{j'-1}, F_M) dF_M\right] dx_{0:j-1}. \tag{5.21}$$

While we could in principle train this model using the ELBO in Eq. (5.20), there is no advantage over the canonical GPSSM model in actually doing so. However, due to the extreme similarity of Eqs. (2.30) and (5.20), an interesting question is whether there is actually a setting of one model in terms of the other model's parameters, such that the ELBOs coincide. This would then allow to interpret one model as the other and use the advantages that either model has to train the respective other one. Intuitively this equivalence of both models can only exist when $R = 1$ since then both describe an equally discretised time-series modelling approach. We summarise our findings about the equivalence between the canonical GPSSM and the SDE GPSSM in the following theorem (Longi et al., 2021):

**Theorem 5.1.** *There exists a setting of parameters for the SDE GPSSM in Eqs.* (5.14) – (5.17) *with* $R = 1$ *and with the variational family in Eq.* (5.19) *such that it has the same optimisation objective as the canonical GPSSM by Doerr et al. (2018) in Eq.* (2.27) *with the residual transition model in Eq.* (5.18) *and the variational family in Eq.* (2.29)*, i.e.,*

$$\mathcal{L}_{SDE} = \mathcal{L}_{PRSSM}, \tag{5.22}$$

*which are given in Eqs.* (2.30) *and* (5.20)*, respectively. This is achieved by setting*

$$k^{\Delta}(x_m^{\Delta}, x_{m'}^{\Delta}) = k(x_m^{\Delta}, x_{m'}^{\Delta}), \qquad k^{\Delta}(x_j, x_{j'}) = k(x_j, x_{j'}) / (R\Delta_t)^2, \tag{5.23}$$

$$k^{\Delta}(x_m^{\Delta}, x_j) = k(x_m^{\Delta}, x_j) / (R\Delta_t), \qquad k^{\Delta}(x_j, x_m) = k(x_j, x_m) / (R\Delta_t), \tag{5.24}$$

$$\sigma_{\Delta}^2 = \sigma_x^2 / (R\Delta_t), \tag{5.25}$$

*and setting all other parameters of the SDE GPSSM, i.e., those of* $p^{\Delta}(x_0), p^{\Delta}(y_j|x_j)$, $p^{\Delta}(F_M)$, $q^{\Delta}(x_0)$, $q^{\Delta}(F_M)$*, the inducing inputs* $X_M^{\Delta}$*, and other parameters of the kernel* $k^{\Delta}$*, to the value of the respective parameters of the canonical GPSSM.*

Note that rescaling the kernel $k^{\Delta}$ in terms of the kernel $k$ in Eqs. (5.23) and (5.24) is often done similarly in multi-output learning. More generally, it can be interpreted as a simple approach to inter-domain GPs (Lázaro-Gredilla and Figueiras-Vidal, 2009). In the following we sketch the proof of Thm. 5.1 which relies heavily on the result of Thm. 4.1 while the full proof is provided in Appx. B.4.

**Sketch of the proof of Theorem 5.1**. The idea of the proof is to show the equivalence in Eq. (5.22)

by demonstrating that all terms in the ELBOs are equal with the settings provided in Thm. 5.1. First, it is easy to see that the KL-terms in Eqs. (2.30) and (5.20) are equal since the respective distributions in both ELBOs are assumed to be the same. It remains to be shown that

$$\sum_{j=1}^{J} \mathbb{E}_{q^{\Delta}(x_j)} \left[ \log p^{\Delta}(y_j|x_j) \right] = \sum_{t=1}^{T} \mathbb{E}_{q(x_t)} \left[ \log p(y_t|x_t) \right].$$

By setting $R = 1$ (as required by Thm. 5.1) we have $J = T/R = T$ and therefore equally many terms in both sums. Additionally the observation model $p^{\Delta}(y_j|x_j)$ is assumed to be the same as $p(y_t|x_t)$. Hence it only remains to be shown that $q^{\Delta}(x_j) = q(x_t)$ for $j = t$. From the general formulas for both terms in Eqs. (2.31) and (5.21), respectively, this is hard to see. Fortunately we already have a formula for $q(x_t)$ with the inducing outputs $F_M$ marginalised out in Thm. 4.1. Obtaining a formula for $q^{\Delta}(x_j)$ in the same way (which we detail in Lem. B.4 in Appx. B.4, where we also provide the proof of the lemma) we can see that the settings in Eqs. (5.23) – (5.25) lead to $q^{\Delta}(x_j)$ and $q(x_t)$ being equal for $j = t$, which completes the proof. □

We therefore have, at least for the standard resolution (corresponding to $R = 1$), two equivalent models: the canonical GPSSM by Doerr et al. (2018) and the SDE GPSSM with the settings as in Thm. 5.1. This finally allows us to do what we set out to achieve at the beginning of this section, namely drawing approximate latent state samples with a constant time requirement for $R > 1$ and $R \in \mathbb{N}$: Instead of using the GPSSM formula for $q(x_t)$ in Eq. (2.31), we can directly use the SDE GPSSM formula for $q^{\Delta}(x_j)$ in Eq. (5.21) to advance time by $R\Delta_t$. The latter can be done in one sampling step, whereas we would always have to sample with the standard resolution $\Delta_t$ with the former approach and therefore need $R$ sampling steps to get from one considered observation to the next. Note that the equivalence of both methods in Thm. 5.1 is only exact for $R = 1$ such that this proposed sampling scheme is an approximation for $R > 1$. However, the low-resolution components (see e.g. Fig. 5.1, right) perform a smoothing of the time series anyway, such that they can also deal with samples of a coarser approximation level.

This completes the description of our multi-resolution GPSSM in Eqs. (5.7) – (5.9) and its efficient training achieved by i) using the backfitting algorithm (Breiman and Friedman, 1985) to efficiently train the different components, ii) employing mini-batches with different resolutions [Eqs. (5.10) and (5.11)], and iii) using Eq. (5.21) to efficiently draw approximate samples for components with low resolutions. The complete algorithm to train this model can be found in Longi et al. (2021, Appx. D) and we continue with the experimental evaluation of our proposed approach in the next section.

## 5.3 Experiments

We validate our multi-resolution GPSSM (MR-GPSSM) on a difficult engine modelling task with more than half a million measurements[4] and on semi-synthetic data derived from this data set. We compare against the approach of Doerr et al. (2018) (PRSSM) and since they already tested against multiple different time-series modelling approaches and achieved favourable results, refrain from doing so again.[5] In order to tease apart the effect of introducing multiple independently trained components versus the different resolutions that we additionally consider, we also compare against a simplification of MR-GPSSM that uses multiple components but all with same resolution which we name multi-component GPSSM (MC-GPSSM). In all experiments we make sure that the different

[4] The dataset is publicly available at https://github.com/boschresearch/Bosch-Engine-Datasets.

[5] Code to reproduce the experiments is available at https://version.helsinki.fi/MUPI/mr-gpssm.

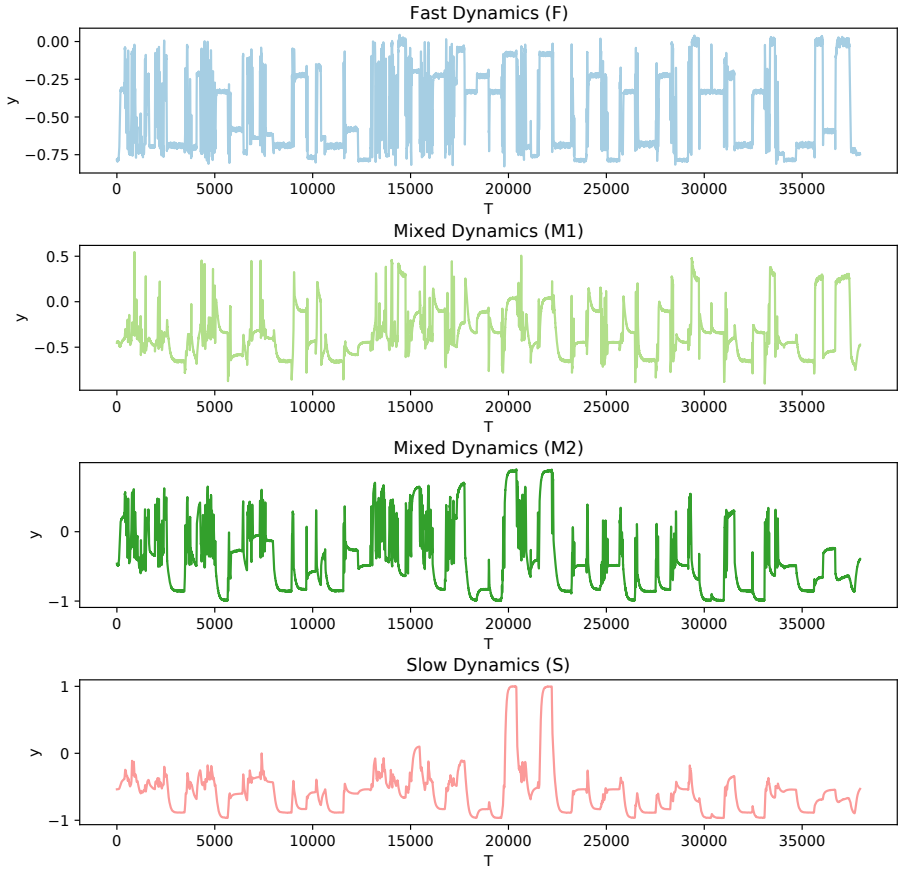


**Figure 5.2:** We depict the semi-synthetic data sets which we use to validate our approach. The one-dimensional time-series with approximately 38000 data points show only fast dynamics, two times mixed (fast and slow) dynamics and only slow dynamics (from top to bottom). Reproduced from Longi et al. (2021).

models get a comparable number of iterations. The exact experimental details such as further model choices and training or model initialisation details can be found in Sec. 4 and Appx. E of Longi et al. (2021). We start by exploring the properties of our new approach on the semi-synthetic data set in Sec. 5.3.1 before applying our method to the full engine modelling task in Sec. 5.3.2

### 5.3.1 Semi-Synthetic Data

We obtain the semi-synthetic data by extracting fast and slowly varying components from the engine data set by training a standard GPSSM capturing either fast components $R^f = 1$ or slow components $R^s = 30$ on different parts of the data and then combining them in different ways (see Longi et al., 2021, Appx. E.2 for a more detailed description). This yields one data set with only fast components (F), two with mixed components (M1, M2) and one with only slow components (S) which are depicted in Fig. 5.2. These data sets offer a perfect playground to compare the different approaches since the optimal resolution is known.

We test the performance of the basic PRSSM approach with the different resolutions $R = 1$ and

**Table 5.1:** We report average RMSEs and their standard errors over five repetitions when applying the models PRSSM (Doerr et al., 2018), MC-GPSSM and MR-GPSSM (our proposed approach) on different semi-synthetic datasets with fast (F), mixed (M1, M2) and slow (S) dynamics (see Fig. 5.2). We use different combinations of the resolutions $R = 1$ and $R = 30$ when trying to model the data. Note that the F dataset is made up only of components with $R = 1$, the S dataset only of those with $R = 30$, while the datasets M1 and M2 contain both resolutions. We mark all methods in bold that perform best or have overlapping confidence intervals with the best performing method. Reproduced from Longi et al. (2022)

| | **PRSSM** | | **MC-GPSSM** | | **MR-GPSSM** |
|---|---|---|---|---|---|
| | $R = 1$ | $R = 30$ | $R = [1, 1]$ | $R = [30, 30]$ | $R = [30, 1]$ |
| **F** | **0.05 (0.00)** | 0.14 (0.00) | **0.06 (0.01)** | 0.16 (0.01) | 0.07 (0.01) |
| **M1** | 0.16 (0.02) | 0.14 (0.00) | 0.15 (0.01) | 0.15 (0.00) | **0.08 (0.01)** |
| **M2** | 0.14 (0.00) | 0.29 (0.11) | 0.14 (0.00) | 0.20 (0.01) | **0.09 (0.01)** |
| **S** | 0.33 (0.08) | **0.16 (0.01)** | 0.29 (0.02) | 0.20 (0.03) | **0.17 (0.02)** |

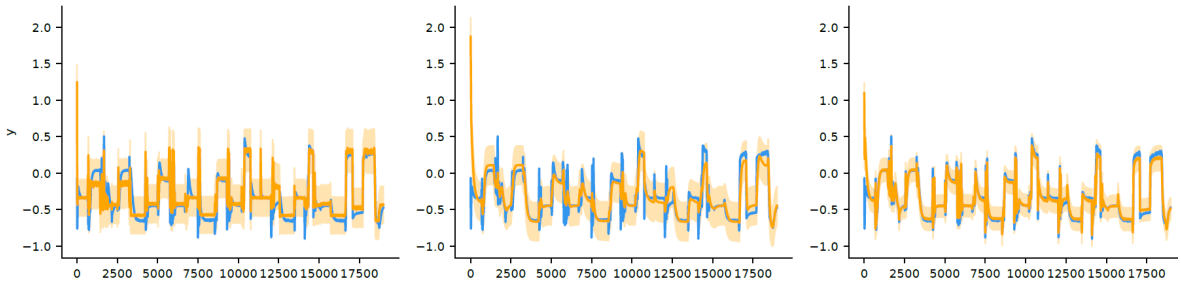


**Figure 5.3:** We show the predictions on the second half of the semi-synthetic dataset with mixed dynamics M1 (see Fig. 5.2) for MC-GPSSM with $R = [1, 1]$ (left), for MC-GPSSM with $R = [30, 30]$ (middle), and for our approach, MR-GPSSM with $R = [30, 1]$ (right). The groundtruth data is shown in blue while the mean of the prediction (solid line) and the confidence intervals (shaded area) are shown in orange. Reproduced from Longi et al. (2021).

$R = 30$, of the MC-GPSSM with two components, once with $R = [1, 1]$ and once with $R = [30, 30]$, and finally our proposed MR-GPSSM approach with two components with different resolutions, $R = [30, 1]$. We report RMSEs when training on the first half of the data and predicting on the second half in Table 5.1. (Test log-likelihood results can be found in Tab. S2 of Longi et al., 2021, Appx. E.2.) We can clearly observe that on the mixed datasets (M1, M2) our proposed multi-resolution approach performs much better than the approaches that only consider a single resolution. This can also be observed visually in Fig. 5.3, where we compare the predictions obtained on dataset M1 for the two MC-GPSSMs with different resolutions and our MR-GPSSM: While MC-GPSSM with the high resolution ($R = [1, 1]$) fails to accurately model slow trends (left) and MC-GPSSM with the low resolution ($R = [30, 30]$) is unable to catch all peaks (middle), our model with $R = [30, 1]$ (right) strikes a very good balance between the other two approaches. (See also Fig. S2 of Longi et al., 2021, Appx. E.2 for the same visual comparisons on the other datasets.) We see furthermore in Table 5.1 that on the fast (F) and slow (S) datasets our approach is on par (for F within two standard errors) with the single-resolution approaches with the correct resolution ($R = 1$ for F and $R = 30$ for S). We therefore conclude that it is important to have a component present in the model that has the correct resolution to accurately represent the data. Finally, there is little difference between the respective single-component (PRSSM) and multi-component (MC-GPSSM) approaches that only use a fixed resolution.

When we discussed why different resolutions are needed in Sec. 5.2, we had mainly two arguments: i) only training with a low resolution when fast dynamic components are present makes it impossible to resolve the latter and ii) only training with a high resolution when slow dynamic components

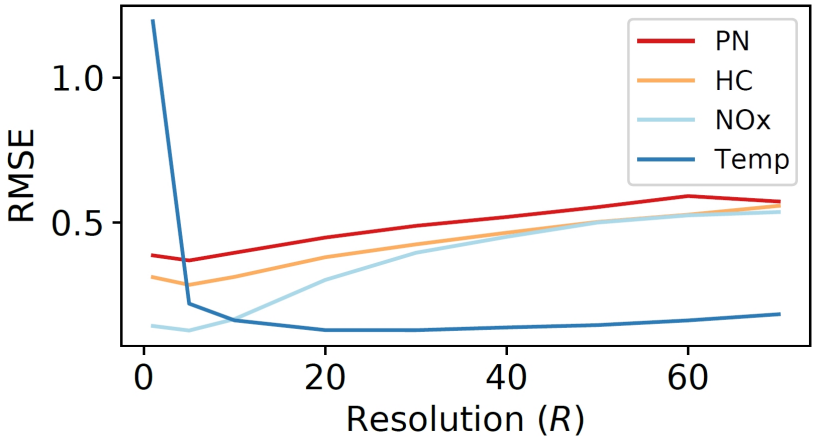


**Figure 5.4:** We show the predictive performance as measured by the RMSE (lower is better) as a function of the parameter $R$ that controls the resolution. Results are for the PRSSM method (Doerr et al., 2018) that is trained independently on the four outputs for each resolution. Reproduced from Longi et al. (2021).

are present (especially when using mini-batches) leads to an insufficiently long history to accurately model long-term trends. We can clearly observe i) qualitatively in Fig. 5.3 (middle) and see it also quantitatively when looking at the high RMSE for the $R = 30$ and the $R = [30, 30]$ models for the F dataset in Table 5.1. For this problem there exists no other solution than to simply use a higher resolution. The problem ii) can equally be observed qualitatively in Fig. 5.3 (left) and quantitatively when looking at the high RMSE for the $R = 1$ and the $R = [1, 1]$ models for the S dataset in Table 5.1. However, for this problem there is at least theoretically a solution, namely simply using a longer history, i.e., larger mini-batches for training. We additionally compare to this approach using the basic PRSSM model: Instead of using a batch with size $B = 50$ (the setting we used for Table 5.1) with resolution $R = 30$, which yields an RMSE of $0.16(0.01)$ within 220 seconds we use $R = 1$ and increase the batch size to $B = 1500$. Training with these settings for the same time leads to unusable predictions since the model is far from being converged, while training for the same number of iterations takes nearly $5000$ seconds and still leads to worse predictions, i.e., an RMSE of $0.30(0.02)$. (See Tab. S3 Longi et al., 2021, Appx. E.2 for further details.) So while being theoretically an alternative we see that this is not a practically viable option.

### 5.3.2 Engine Modelling Task

Having explored some aspects of our proposed approach we are finally ready to apply it to the real-world engine modelling task. The data set consists of 22 independent measurements which we split into 16 train and 6 test data sets. Each of these has four outputs, particle numbers (PN), hydrocarbon concentration (HC), nitrogen oxide concentration (NOx), and the engine temperature (Temp), which we model independently. (See Longi et al., 2021, Appx. E.3 for further details concerning the data set.) We first perform a grid search to determine the optimal resolutions for the basic PRSSM approach (Doerr et al., 2018) by setting $R$ to several values between 1 and 70, training the models and recording the predictive performance. The results for the RMSE can be seen in Fig. 5.4 and show that while a high resolution ($R$ between 1 and 5) is optimal for PN, HC, and NOx, the Temp data set requires a much lower resolution ($R$ around 30) for optimal performance. Therefore, we decide to use a combination of these resolutions for our experiment with MR-GPSSM for which we choose $R = [30, 5, 1]$. We compare against the performance of PRSSM with $R \in \{30, 5, 1\}$ and the corresponding MC-GPSSMs that use three components with the same resolution. The results of this comparison are depicted in Fig. 5.5. We observe that our approach is competitive on each of

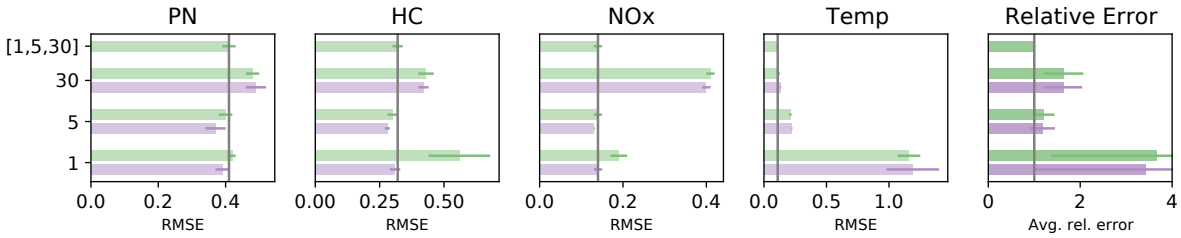


**Figure 5.5:** The four leftmost plots show the predictive performance as measured by the RMSE (lower is better) for different models and resolutions on the four outputs PN, HC, NOx, and Temp. Results are mean value and standard error over five repetitions. The single component method (PRSSM Doerr et al., 2018) is shown in purple while the multi-component methods (our method MR-GPSSM on top, and MC-GPSSM for different resolutions) are shown in green. For PRSSM and MC-GPSSM we present results for different resolutions where MC-GPSSM always uses three components with the same resolution $R \in \{30, 5, 1\}$. The grey vertical line eases the visual comparison with our method. In the rightmost plot we show the relative error (with respect to our method) averaged over the four other plots. Reproduced from Longi et al. (2022).

**Table 5.2:** We examine the robustness of our MR-GPSSM approach for different combinations of the resolutions of the multiple components. Reported are average RMSE values (the lower the better) and their standard errors over five repetitions for the four outputs over four different combinations of resolutions. The first data column ($R = [30, 5, 1]$) corresponds to the setting used in Fig. 5.5. Reproduced from Longi et al. (2021).

| Output \ R | $[30, 5, 1]$ | $[7, 5, 1]$ | $[40, 20]$ | $[40, 20, 10, 5, 1]$ |
|---|---|---|---|---|
| PN | $0.41(0.02)$ | $0.40(0.03)$ | $0.45(0.02)$ | $0.39(0.02)$ |
| HC | $0.32(0.02)$ | $0.29(0.01)$ | $0.40(0.02)$ | $0.32(0.01)$ |
| NOx | $0.14(0.01)$ | $0.15(0.01)$ | $0.33(0.01)$ | $0.17(0.02)$ |
| Temp | $0.11(0.01)$ | $0.22(0.03)$ | $0.11(0.00)$ | $0.10(0.00)$ |

the outputs while the single-resolution methods PRSSM and MC-GPSSM only perform well if the resolution is adequately set. Their performance is especially bad on the output Temp with a very high resolution ($R = 1$) or on NOx with a very low resolution ($R = 30$) in accordance with the findings in Fig. 5.4. A further general observation that we make is that the RMSEs for the particle numbers (PN) are generally higher than for the other outputs which is in line with observations from the literature (e.g. Frommater, 2018). Comparing the average relative error with respect to the performance of our method across the four outputs (Fig. 5.5, right), we see that our multi-resolution approach has slight advantages compared with the other approaches that only use a single resolution. We attribute this good performance to the robustness that MR-GPSSM achieves over multiple data sets on which the single-resolution approaches need widely different resolutions for optimal performance. In our final experiment we check whether this robustness also holds across different sets of resolutions when using our MR-GPSSM approach. In Table 5.2 we compare the performance of the resolution which we used to produce the results in Fig. 5.5 (first column) with one example where only high resolutions (second column), one where only low resolutions (third column), and another one where mixed resolutions (last column) have been used. (See Tab. S5 in Longi et al., 2021, Appx. E.3 for an extended comparison.) From this we can conclude that for the Temp output a low-resolution component is definitely needed while NOx and HC require a high-resolution component. Otherwise the results are very robust over different combinations of resolutions.

## 5.4 Chapter Summary

In the current chapter we presented a new approach for variational inference in GPSSMs that allows to traverse time with different resolutions. Our model consists of multiple components that evolve

independently over time which allows us to use different resolutions for the different components and also to train them efficiently employing the backfitting algorithm (Breiman and Friedman, 1985). We further increased the efficiency of our approach by exploiting a connection to stochastic differential equations (SDEs). This allowed us to sample approximate subsequences of the same length for components with different resolutions (which therefore have different history lengths) at the same computational cost. We empirically validated on semi-synthetic data that our approach outperforms models that allow only for a single resolution when there are multiple components with different resolutions present in the data. On semi-synthetic data with only one resolution and on a challenging real-world engine modelling data set our approach was on par with the single-resolution approaches that were using an optimised resolution. However, our model proved robust across different tasks using always the same set of resolutions. This avoids having to find an optimal resolution first which is necessary for the single-resolution models.

Our presented method is general and can also be used in combination with different approaches to inference in GPSSMs. While we based our approach on the model of Doerr et al. (2018) in this chapter, the former is equally applicable to other GPSSM approaches (e.g. Eleftheriadis et al., 2017; Ialongo et al., 2019 or the one by Lindinger et al., 2022 which we present in detail in the following chapter). Similar to other GPSSMs our model is also applicable to other fields than the automotive area such as e.g. medicine (e.g. Lipton et al., 2016), human motion prediction (e.g. Martinez et al., 2017) or neuroscience (e.g. Prince et al., 2021).

An interesting direction for future work could be to further explore the connection to SDEs via other works that also use them in combination with GPs (e.g. Ruttor et al., 2013; Yıldız et al., 2018; Hegde et al., 2019; Zhao et al., 2020) and thus probing new ways to make inference even more efficient and precise.

CHAPTER 6

# Laplace Approximated Gaussian Process State-Space Models

Uncertainty estimation in time-series modelling (see e.g. Särkkä, 2013) is a hard task since two different noise sources have to be taken into account: First, the observation noise that stems from possibly noisy measurements of the system under consideration and second, the process noise associated with the uncertain development of the system. Especially the second noise source leads to an accumulation of uncertainty over time and renders predictions far into the future difficult (see also Sec. 1.2.3).

Gaussian processes (Rasmussen and Williams, 2006) provide a well established framework for dealing with uncertainty (see Sec. 2.1 for some technical background). They define a distribution over functions $f \sim p(f)$ and can be used as building blocks in state-space models for modelling complex temporal dependencies. This model class, aptly called Gaussian process state-space models (Frigola, 2015, see also Sec. 2.4), can be used to model observations $y_t \in \mathbb{R}^{d_y}$ from a time series, where $t = 1, \ldots, T$ is the time index. It assumes that each observation $y_t$ is emitted by a latent state $x_t \in \mathbb{R}^{d_x}$, $y_t \sim p(y_t|x_t)$, and that the latent states have a Markovian structure, i.e., $x_{t+1} \sim p(x_{t+1}|x_t, f)$. The noise of the latter model (called transition model) and the function uncertainty of the Gaussian process are propagated over time, resulting in complex and non-Gaussian behaviour. The flexibility of this model class paired with its probabilistic predictions makes it an interesting building block e.g. for model-based reinforcement learning (e.g Deisenroth and Rasmussen, 2011).

Since the first work on Gaussian process state space models (Wang et al., 2005), much work has been devoted to deriving efficient and accurate inference schemes. The arguably most expressive model at this point is by Ialongo et al. (2019) which assumes a flexible, parametric Markov-structured Gaussian posterior over the temporal states, $x_t \sim q(x_t|\cdot)$, and allows for dependencies with the Gaussian process posterior. [See Sec. 2.4, especially Eqs. (2.26) and (2.34) for a description of their approach.] However, its empirical performance often does not match its theoretical expressiveness: In the original publication the authors report many cases in which this model is outperformed by an easier alternative that simply sets $q(x_t|\cdot)$ to the prior transition (Doerr et al., 2018). [See also Sec. 2.4, especially Eqs. (2.27) and (2.29) for a description of their approach to inference in GPSSMs.] The gap between the empirical and theoretical performance of Ialongo et al. (2019) can most likely be attributed to the hard learning problem created by employing a flexible, parametric $q(x_t|\cdot)$: The additional parameters of such an approach have to be estimated, and, moreover, one can only start optimizing the parameters governing the temporal state $x_t$ once the parameters governing $x_{t-1}$ begin converging since the inference scheme is built on sequential sampling of the temporal states (see Sec. 2.4 for a detailed discussion).

In this chapter we address these issues by presenting a novel inference algorithm for Gaussian process state-space models (Lindinger et al., 2022) that solves the latter problem and explores the trade-off between the very expressive approach by Ialongo et al. (2019) and the method by Doerr et al. (2018) which has good optimisation properties due to its parameter-free (but simple) choice for $q(x_t|\cdot)$.[1] Our approach applies stochastic variational inference over the Gaussian process posterior

[1] Note that we ruled out two other algorithmic changes introduced by Ialongo et al. (2019) (compared to Doerr et al., 2018) as being not relevant for major changes in performance in Chap. 4 (see especially Sec. 4.3).

(Hensman et al., 2013) and, conditioned on it, the Laplace approximation (see e.g. MacKay, 2003) over the temporal states, thereby allowing for complex dependencies. Inference is performed via a double loop algorithm in which we optimise over the Gaussian process posterior in the outer loop and over temporal states in the inner loop. The resulting approximate posterior over the temporal states, $q(x_t|\cdot)$, has a Markov Gaussian form and is found by a joint optimisation over all temporal latent states. The latter addresses the issue of Ialongo et al., 2019, in which the parameters governing $x_t$ can only be learned once the estimate of the previous state, $x_{t-1}$, is meaningful. In order to arrive at a computationally efficient approach, we (i) compute cheap gradients through the Laplace approximation by using the inverse function theorem (see e.g. Krantz and Parks, 2002), (ii) exploit the Markovian structure of our model (see e.g. Bell, 2000), (iii) approximate the Gaussian process posterior using inducing points (Quinonero-Candela and Rasmussen, 2005) and (iv) apply mini-batching. Our experiments confirm the benefits of this novel inference scheme which provides higher quality uncertainty estimates than its state-of-the-art alternatives.

The remainder of this chapter is structured as follows. First, in Sec. 6.1, we provide additional background on the Laplace approximation and discuss its differences to variational inference, while we already presented the background on GPSSMs in detail in Sec. 2.4. Next, we introduce our new method and discuss different improvements that are needed in order to make it efficient (Sec. 6.2). In Sec. 6.3, we relate our approach to other works in the literature, going beyond the brief introductions in Sec. 2.4 by pointing out similarities and differences. Then, we experimentally compare our approach to the methods of Doerr et al. (2018) and Ialongo et al. (2019) on a toy data set and on several benchmark data sets in Sec. 6.4. Finally, in Sec. 6.5, we summarise the current chapter.

## 6.1 The Laplace Approximation

In this section we first introduce the Laplace approximation by taking its existing application to parametric state-space models (e.g. Skaug and Fournier, 2006) as an example. Then, in the second part, we discuss similarities and differences between the Laplace approximation and variational inference which we employed in the previous chapters.

### 6.1.1 Laplace Approximated Parametric State-Space Models

As we already briefly discussed in Sec. 1.2.3, the idea of using (parametric) state-space model is ubiquitous in the statistical and machine learning community (see e.g. Särkkä, 2013) for time-series modelling and goes back a long time (e.g. Kalman, 1960). The work by Skaug and Fournier (2006) proposes to use the Laplace approximation for inference in state-space models and a brief discussion of this approach gives us the perfect opportunity to introduce the Laplace approximation. We start this discussion with a brief reminder about parametric state-space models.

State-space models (see e.g. Särkkä, 2013) offer a principled way to model time series, i.e., noisy observations $Y_T = \{y_t\}_{t=1}^T$ from a dynamical system, where $t$ is the time index. In order to disentangle the dynamics from the observational (or measurement) noise, state-space models use latent states $X_{T_0} = \{x_t\}_{t=0}^T$ that are then assumed to form a Markov sequence. Such a model is completely described by the *initial distribution* $p_\theta(x_0)$, a *transition model* $p_\theta(x_t|x_{t-1})$ and the *emission model* $p_\theta(y_t|x_t)$, resulting in

$$p_\theta(Y_T, X_{T_0}) = p_\theta(x_0) \prod_{t=1}^{T} p_\theta(y_t|x_t) p_\theta(x_t|x_{t-1}), \tag{6.1}$$

where generally the transition and emission model, and the initial distribution depend on model pa-

rameters $\theta$ that we wish to learn.[2] This learning is typically performed by maximising the marginal likelihood of the observations $Y_T$ under our model (also called evidence) [cf. Eq. (2.5) for the equivalent quantity for GP regression],

$$L(\theta) = p_\theta(Y_T) = \int p_\theta(Y_T, X_{T_0}) dX_{T_0}, \tag{6.2}$$

with respect to $\theta$.[3] With the notable exception of linear Gaussian state-space models (Kalman, 1960), computing Eq. (6.2) is not analytically possible and we have to resort to approximations.

In the following, we introduce the Laplace approximation (see e.g. MacKay, 2003; Skaug and Fournier, 2006) which can be used to obtain an approximate maximum likelihood estimate of the parameters $\theta$ by approximating $L(\theta)$ in Eq. (6.2). In order to get a qualitative feeling of how the Laplace approximation works in practice, see Fig. 6.1 where we show the approximations to different one-dimensional probability density functions (grey) using the Laplace approximation (brown). For a quantitative treatment, we start by introducing $g$ as a short-hand for the log-joint:

$$g(X_{T_0}, \theta) \equiv \log p_\theta(Y_T, X_{T_0}), \tag{6.3}$$

(note that $Y_T$ is constant and hence not considered as a variable). We denote its maximiser[4] with respect to (wrt.) the latent states as

$$\hat{X}_{T_0} = \operatorname{argmax}_{X_{T_0}} g(X_{T_0}, \theta). \tag{6.4}$$

Performing a second-order Taylor approximation of $g(X_{T_0}, \theta)$ in Eq. (6.3) around this mode $\hat{X}_{T_0}$, plugging the (exponentiated) result back into Eq. (6.2), and then evaluating the resulting Gaussian integral yields (see Appx. A.3.1 for a detailed derivation)

$$L(\theta) \approx \widetilde{p}_\theta(Y_T) = \sqrt{2\pi}^{d_x(T+1)} p_\theta(Y_T, \hat{X}_{T_0}) \det\left(H(\theta)\right)^{-\frac{1}{2}}, \tag{6.5}$$

$$H(\theta) = -\frac{\partial^2}{\partial X_{T_0}^2} g(X_{T_0}, \theta)\bigg|_{X_{T_0}=\hat{X}_{T_0}}. \tag{6.6}$$

Here and in the following we use $\widetilde{p}$ to denote a distribution, where the Laplace approximation has been applied to $X_{T_0}$ (and which therefore depends on $\hat{X}_{T_0}$ in Eq. (6.4)). The expression in Eq. (6.5) can then be optimised numerically wrt. $\theta$ in order to estimate the parameters $\theta^* = \operatorname{argmax}_\theta \widetilde{p}_\theta(Y_T)$ (Skaug and Fournier, 2006). Note that this can be done efficiently since the Hessian $H$ in Eq. (6.6) is sparse and structured which we discuss in detail in Sec. 6.2.3. The same methodology can also be efficiently applied to other latent variable models with a sparse or structured Hessian (e.g. Rue et al., 2009; Kristensen et al., 2016).

### 6.1.2 Laplace Approximation versus Variational Inference

Before we finally present our proposed method that combines the approximate inference approaches variational inference (that we introduced in detail in Appx. A.2) and the Laplace approximation (pre-

[2] Note that this was generally also the case for GPSSMs [see e.g. Eq. (2.25)] but as we shall see in Sec. 6.2, the dependence on model parameters is particularly important in this chapter which we mark by using $p_\theta$.

[3] A fully Bayesian approach also treats the parameters $\theta$ probabilistically and then aims at estimating its posterior. See e.g. Rue et al. (2009) for an approach that does this approximately using the Laplace approximation.

[4] This is in general a local optimum as the optimisation problem is non-convex. However, if the observations are dense, the locally linear approximation to the dynamics made implicitly by the Laplace approximation is a reasonable assumption (see e.g. Eleftheriadis et al., 2017) leading to well-identifiable optima.

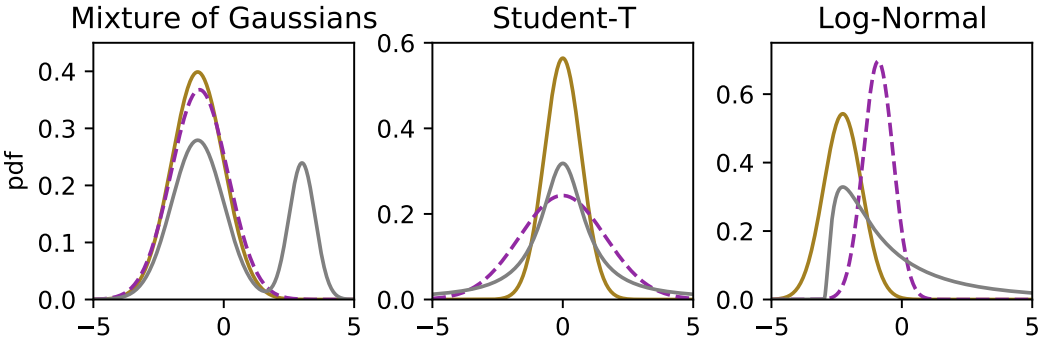


**Figure 6.1:** We show the probability density functions (pdfs) of three one-dimensional distributions (Gaussian mixture, student's-t, and log-normal distribution from left to right) in grey. Furthermore, we depict the optimal (numerical) approximations to these groundtruth distributions using the Laplace approximation (brown, solid) and variational inference with a Gaussian variational family (magenta, dashed).

sented in the previous section), we provide a short comparison of them. Using variational inference with a Gaussian variational family leads to a similar approach to the Laplace approximation since both methods use the same functional form for the approximate distribution and both approaches are mode seeking (see Fig. 6.1, left). While the Laplace approximation fits the mean to a mode of the distribution and has the same curvature as the true function at this mode, variational inference minimises the KL-divergence between the approximate and the true distribution. Due to the KL-divergence being heavily penalised by placing mass of the approximating distribution in regions that have zero mass under the true distribution [see Eq. (A.6)], variational inference avoids this (see Fig. 6.1, right). Both approximations are not ideally suited to approximating heavy-tailed distributions due to their Gaussianity assumptions, but the Laplace approximation even slightly less so since matching the curvature at the mode typically leads to narrower distributions in this case (see Fig. 6.1, middle).

However, as we introduce next, in our model we only use the Laplace approximation for approximating the posterior over the temporal states. For many real-world applications, the dynamics can often be well described locally by a linear model, justifying the Gaussian approximation (e.g. Eleftheriadis et al., 2017).

## 6.2 Combining Variational Inference and the Laplace Approximation

In this section we propose a new inference method for the GPSSM in Eq. (2.27), i.e., the model used in Doerr et al., 2018 that we repeat here for convenience:

$$p_\theta(Y_T, X_{T_0}, F_M) = p_\theta(x_0)p_\theta(F_M)\prod_{t=1}^{T} p_\theta(y_t|x_t)p_\theta(x_t|x_{t-1}, F_M). \tag{6.7}$$

In contrast to the parametric state-space model in Eq. (6.1), this model has an additional set of latent variables, the inducing outputs $F_M$ that describe the behaviour of the GP in the transition model.[5] Instead of relying solely on variational methods for performing inference over both sets of latent

[5] Note that we use the model of Doerr et al. (2018) here that relies on the FITC approximation which is criticised by Ialongo et al. (2019). However, since we showed in detail in Sec. 4.2 that the latter work could have used the FITC approximation as well without changing the optimisation objective, we rely on it here since it simplifies some calculations. Furthermore, we will later use an approximate posterior over $F_M$ [Eq. (6.14)] that counteracts the problematic behaviour of vanishing noise variances that this approximation can entail (see also footnote 14 in Chap. 2).

variables as previous works on inference in GPSSMs (e.g. Frigola et al., 2014; Eleftheriadis et al., 2017; Ialongo et al., 2019) have done, we employ variational inference in combination with the Laplace approximation. This allows us to treat the local latent variables, i.e. the temporal states $X_{T_0}$, and the global latent variables, i.e. the inducing outputs $F_M$, differently. Note that naively applying the Laplace approximation to this model, i.e., without making a distinction between the different sets of latent variables, (i) does not lead to an efficient algorithm since the resulting Hessian would not have an exploitable sparsity structure (see Sec. 6.2.3), (ii) would assume a linear relationship between the two latent variable classes and (iii) would make the Gaussian assumption made by the Laplace approximation more questionable.

We start by deriving our optimisation objective in Sec. 6.2.1 and then, in Sec. 6.2.2, discuss a caveat that we encounter when trying to naively optimise this objective. Afterwards, we show how the sparsity of the Hessian in the Laplace approximation can be exploited for computational savings (Sec. 6.2.3) before summarising the algorithm in Sec. 6.2.4.

### 6.2.1 Optimisation Objective

Similarly as the previous methods, we wish to find an approximation to the log marginal likelihood $\log p_\theta(Y_T)$. We start from its definition given the model under consideration, [Eq. $(6.7)$],

$$\log p_\theta(Y_T) = \log \int p_\theta(Y_T|F_M)p_\theta(F_M)dF_M. \tag{6.8}$$

Here, we introduced $p_\theta(Y_T|F_M) = \int p_\theta(Y_T, X_{T_0}|F_M)dX_{T_0}$, where the integrand can be obtained from Eq. $(6.7)$ using $p_\theta(Y_T, X_{T_0}|F_M) = p_\theta(Y_T, X_{T_0}, F_M)/p_\theta(F_M)$. Applying the Laplace approximation (Sec. 6.1) to the latter integral and plugging it in Eq. (6.8) leads to

$$\log p_\theta(Y_T) \approx \log \int \widetilde{p}_\theta(Y_T|F_M)p_\theta(F_M)dF_M. \tag{6.9}$$

Here, $\widetilde{p}_\theta(Y_T|F_M)$ is the Laplace approximation of the conditional $p_\theta(Y_T|F_M)$, which is given by [cf. Eq. $(6.5)$]

$$\widetilde{p}_\theta(Y_T|F_M) = \sqrt{2\pi}^{d_x(T+1)} p_\theta(Y_T, \hat{X}_{T_0}|F_M) \det\left(H(\theta, F_M)\right)^{-\frac{1}{2}}, \tag{6.10}$$

where $\hat{X}_{T_0}$ [cf. Eq. (6.4)] is a mode of the log-density [cf. Eq. (6.3)]

$$g_{\text{GP}}(X_{T_0}, \theta, F_M) = \log\left[p_\theta(x_0)\prod_{t=1}^{T} p_\theta(y_t|x_t)p_\theta(x_t|x_{t-1}, F_M)\right] \tag{6.11}$$

and

$$H(\theta, F_M) = -\frac{\partial^2}{\partial X_{T_0}^2} g_{\text{GP}}(X_{T_0}, \theta, F_M)\Big|_{X_{T_0}=\hat{X}_{T_0}} \tag{6.12}$$

is the corresponding Hessian [cf. Eq. $(6.6)$].

We proceed by using the methodology from variational inference to lower bound the resulting expression in Eq. $(6.9)$: First, we multiply the term within the integral by a suitably chosen one,

$$\log p_\theta(Y_T) \approx \log \int q_\psi(F_M)\frac{\widetilde{p}_\theta(Y_T|F_M)p_\theta(F_M)}{q_\psi(F_M)}dF_M, \tag{6.13}$$

where $q_\psi(F_M)$ is the variational distribution, an arbitrary distribution over the $F_M$ parameterised by

$\psi$. Finally, we use Jensen's inequality to push the logarithm inside of the resulting integral (thereby lower bounding the expression) in Eq. (6.13), and use the definition of the KL-divergence [Eq. (A.6)] to arrive at our optimisation objective,

$$\mathcal{L}(\theta, \psi) \equiv \int q_\psi(F_M) \log \widetilde{p}_\theta(Y_T|F_M) dF_M - KL(q_\psi(F_M) \parallel p_\theta(F_M)), \tag{6.14}$$

which is an approximate lower bound for the log evidence in Eq. (6.8), i.e., it obeys $\log p_\theta(Y_T) \gtrapprox \mathcal{L}(\theta, \psi)$. There are two things to note abound this bound: First, it is an *approximate* lower bound, since the Laplace approximation does not provide a valid bound but only an approximation. Second, the property that we minimise a KL-divergence between the true posterior $p_\theta(F_M|Y_T)$ and the approximate posterior $q_\psi(F_M)$ by optimising this bound (see Sec. A.2), also only holds approximately. In fact, the KL-divergence that is minimised by optimising this bound is $KL\left[q_\psi(F_M) \parallel \widetilde{p}_\theta(F_M|Y_T)\right]$, i.e., $q_\psi(F_M)$ approximates the posterior after applying the Laplace approximation to the latent states $X_{T_0}$.

In principle, we could now go ahead, choose a parametric family for $q_\psi(F_M)$, evaluate our bound in Eq. (6.14), and use automatic differentiation to optimise the parameters $\theta$ and $\psi$. However, this is inefficient for two reasons: First, evaluating Eq. (6.10) involves an optimisation to obtain $\hat{X}_{T_0}$ and automatic differentiation through this optimisation is inefficient. Second, we need the determinant of the Hessian $H(\theta, F_M)$ [Eq. (6.12)] in order to evaluate Eq. (6.10) which scales cubically in the size of the Hessian. In the following two sections we provide solutions to both of these efficiency problems.

### 6.2.2 Implicit Function Theorem

Next, we turn to an important dependence in our construction: The mode $\hat{X}_{T_0}$ depends on the setting of the model parameters $\theta$. Since our optimisation objective $\mathcal{L}(\theta, \psi)$ in Eq. (6.14) involves the mode $\hat{X}_{T_0}$ of the log-density $g_{\text{GP}}(X_{T_0}, \theta, F_M)$ [Eq. (6.11)], we require the derivative $\frac{\partial \hat{X}_{T_0}}{\partial \theta}$ in order to compute $\frac{\partial \mathcal{L}}{\partial \theta}$.[6] Being used to automatic differentiation we would usually leave this calculation to our favourite framework, since obtaining the mode $\hat{X}_{T_0}$ is nothing but a (rather long) sequence of summations and multiplications which automatic differentiation can deal with. Nevertheless, since several optimisation steps are needed to compute the mode, back-propagating through optimisation would lead to a large memory footprint and long execution times. Instead, we can calculate the derivative of the mode wrt. $\theta$ solely with the value of $\hat{X}_{T_0}$, i.e., independent of the steps taken to get there, with the help of the implicit function theorem [IFT, see e.g. Krantz and Parks, 2002].[7] Summarising the problem, we have our optimisation objective $\mathcal{L}(\theta, \psi)$ [Eq. (6.14)] which depends [through Eq. (6.10)] on the mode $\hat{X}_{T_0}$ of the log-density $g_{\text{GP}}(X_{T_0}, \theta, F_M)$ [Eq. (6.11)]. Since $g_{\text{GP}}$ is a function of $\theta$, there will be a dependence of $\hat{X}_{T_0}$ on $\theta$, which is why we require $\frac{\partial \hat{X}_{T_0}}{\partial \theta}$ in order to compute $\frac{\partial \mathcal{L}}{\partial \theta}$. In Appx. A.3.2, we show that the former derivative can be obtained as

$$\frac{\partial \hat{X}_{T_0}(\theta)}{\partial \theta} = -H^{-1}(\theta, F_M) \frac{\partial h(\hat{X}_{T_0}, \theta, F_M)}{\partial \theta}, \tag{6.15}$$

[6] Through the $F_M$, there is also a dependence on $\psi$ which must be treated in the same way. We omit the equivalent derivation for the sake of clarity and conciseness.

[7] The website http://implicit-layers-tutorial.org/ provides an introduction to the IFT geared to the Machine Learning practitioner.

which we derive using the IFT and where

$$h(x, \theta, F_M) = -\frac{\partial \log p_\theta(Y_T, X_{T_0} \mid F_M)}{\partial X_{T_0}}\bigg|_{X_{T_0}=x} \tag{6.16}$$

is the Jacobian of the function $g_{\text{GP}}$ [Eq. (6.11)] (see also Skaug and Fournier, 2006). Both terms on the right hand side of Eq. (6.15) can be obtained using automatic differentiation and require only the value of $\hat{X}_{T_0}$ such that the complete computational graph of how it has been obtained is no longer required. Note that Eq. (6.15) exchanges potentially costly automatic differentiation computations with a Hessian solve. Naively, this would incur memory and time costs scaling quadratically and cubically in the size of the latent state $X_{T_0}$, respectively. Therefore, this does only lead to an efficient algorithm by also exploiting the structure and sparsity of the Hessian, as we will discuss next.

### 6.2.3 Sparsity and Structure of the Hessian

Taking a closer look at the definition of the Hessian $H(\theta, F_M)$ in Eq. (6.12) that is needed for Eqs. (6.10) and (6.15), we realise that it is given as the second partial derivatives of a sum of $T+1$ terms since $g_{\text{GP}}$ in Eq. (6.11) is given as the logarithm of a product of $T+1$ terms:

$$H(\theta, F_M) = -\frac{\partial^2}{\partial X_{T_0}^2}\left[\log p_\theta(x_0) + \sum_{t=1}^{T} \left(\log p_\theta(y_t|x_t) + \log p_\theta(x_t|x_{t-1}, F_M)\right)\right].$$

Due to the Markovian structure of our model, all second partial derivatives wrt. latent states being more than one time step $t$ apart vanish, i.e.,

$$H_{tt'}(\theta, F_M) = -\frac{\partial^2 g_{\text{GP}}(X_{T_0}, \theta, F_M)}{\partial x_t \partial x_{t'}} = 0 \quad \text{for} \quad t' \notin \{t-1, t, t+1\}.$$

This results in a block-tridiagonal structure of the Hessian:

$$H = \begin{pmatrix} A_0 & B_1 & 0 & \cdots & 0 \\ B_1^\top & A_1 & B_2 & \ddots & \vdots \\ 0 & B_2^\top & A_2 & B_3 & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & B_T^\top & A_T \end{pmatrix}, \quad A_t = -\frac{\partial^2 g_{\text{GP}}}{\partial x_t \partial x_t}, \quad B_t = -\frac{\partial^2 g_{\text{GP}}}{\partial x_t \partial x_{t-1}}. \tag{6.17}$$

with $A_t, B_t \in \mathbb{R}^{d_x \times d_x}$ such that $H \in \mathbb{R}^{d_x(T+1) \times d_x(T+1)}$. This structure can also be found in similar models (see e.g. Bell, 2000) and the recent work by Durrande et al., 2019 considers the efficient implementation of computations for similar structures (banded matrices) into automatic differentiation frameworks.

The structure in Eq. (6.17) reveals another interesting aspect of our algorithm: The first step of the Laplace approximation consists of making a multivariate Gaussian approximation to the posterior $p_\theta(X_{T_0}|Y_T, F_M)$, where $\hat{X}_{T_0}$ is the mean and $H(\theta, F_M)$ is the precision matrix (see Appx. A.3.1, especially Eq. (A.37)). Therefore, the structure of $H$ tells us something about the underlying (implicit) structural assumption that we have used to approximate $p_\theta(X_{T_0}|Y_T, F_M)$. Exploiting the structure in Eq. (6.17) and using standard formulas for Gaussian conditionals (Toussaint, 2011, Eq. (43)), we can

rewrite the approximate posterior to find a linear Markov Gaussian model,

$$\mathcal{N}\left(X_{T_0}\middle|\hat{X}_{T_0}, H^{-1}\right) = \prod_{t=0}^{T} \mathcal{N}\left(x_t | a_t + b_t x_{t-1}, c_t\right), \tag{6.18}$$

where the coefficients $a_t$, $b_t$, and $c_t$ depend on $\hat{X}_{t-1:T}$ and the blocks $A_{t:T}$ and $B_{t:T}$ of the Hessian $H$ (here we used the shorthand $A_{t:T} = \{A_{t'}\}_{t'=t}^{T}$). Such a linear Markov Gaussian model is also used by Eleftheriadis et al. (2017), where the authors use parameters $a_t$, $b_t$, and $c_t$ that have to be optimised during inference while the conditional dependence on the $F_M$ is not taken into account.

Turning our attention back to the structure revealed in Eq. (6.17), there are three aspects of the algorithm for which we can achieve considerable computational savings when we take it into account: i) obtaining the blocks $A_t$ and $B_t$ [Eq. (6.17)] of the Hessian while not calculating the unnecessary zero blocks, ii) calculating the determinant of the Hessian required for Eq. (6.10), and finally iii) performing the Hessian solve in Eq. (6.15). Problem i), while not hard, requires a technical solution which we detail in Appx. A.3.3. Problems ii) and iii) can both be solved by following e.g. Koulaei and Toutounian (2007) in noting that the Hessian $H$ in Eq. (6.17) allows a factorisation that can be exploited: It is

$$H = (\Lambda + B^\top)\Lambda^{-1}(\Lambda + B), \tag{6.19}$$

where $B$ is the strictly upper-triangular block matrix consisting only of the blocks $B_t$ in Eq. (6.17) and $\Lambda$ is a block diagonal matrix that can be calculated from the blocks $A_t$ and $B_t$. We provide a detailed derivation and show how Eq. (6.19) can be used in order to solve problems ii) and iii) in Appx. A.3.4. Implementing these improvements leads to a reduction of the memory footprint of the algorithm from $\mathcal{O}(T^2 d_x^2)$ to $\mathcal{O}(T d_x^2)$ [mainly through i)] and a reduction of the theoretical runtime from $\mathcal{O}(T^3 d_x^3)$ to $\mathcal{O}(T d_x^3)$ through ii) and iii).

### 6.2.4 Algorithm

**Algorithm 1** Optimisation objective for Laplace approximated GPSSMs

Given time series $Y_T$
Choose number of iteration and samples $I$ and $N$, latent state dimension $d_x$
Initialise model parameters $\theta$, variational parameters $\psi = \{m, S\}$
**for** $i = 1 \dots I$ **do** ▷ Optimisation steps
  **for** $n = 1 \dots N$ **do** ▷ Reparameterised samples
    Sample $F_M^{(n)} \sim q_\psi(F_M)$
    Find $\hat{X}_{T_0}^{(n)}$ by maximising $g_{\text{GP}}(X_{T_0}, \theta, F_M^{(n)})$ ▷ Eq. (6.11)
    Obtain non-zero elements of $H$ $(A_t, B_t)$ ▷ Appx. A.3.3
    Evaluate $\det H$ ▷ Eqs. (A.43) and (A.44)
    Evaluate $\widetilde{p}_\theta(Y_T \mid F_M^{(n)})$ ▷ Eq. (6.10)
  **end for**
  Evaluate $\mathcal{L}(\theta, \psi)$ ▷ Eqs. (6.14) and (6.20)
  Obtain gradients $\frac{\partial \mathcal{L}}{\partial \theta}$, $\frac{\partial \mathcal{L}}{\partial \psi}$, using custom $\frac{\partial \hat{X}_{T_0}^{(n)}}{\partial \theta}$, $\frac{\partial \hat{X}_{T_0}^{(n)}}{\partial \psi}$ ▷ Eqs. (6.15) and (A.45)
  Update $\theta$ and $\psi$
**end for**

In order to evaluate and maximise our optimisation objective $\mathcal{L}(\theta, \psi)$ in Eq. (6.14), we need to choose a parametric family for $q_\psi(F_M)$. We follow the literature (e.g. Ialongo et al., 2019) and take a

Gaussian distribution $q_\psi(F_M) = \mathcal{N}(F_M|m, S)$, allowing an analytical evaluation of the KL-term in Eq. (6.14). The first term on the right hand side of Eq. (6.14) is analytically intractable so we resort to sampling,

$$\int q_\psi(F_M) \log \widetilde{p}_\theta(Y_T|F_M) dF_M \approx \sum_{n=1}^{N} \log \widetilde{p}_\theta(Y_T|F_M^{(n)}), \tag{6.20}$$

with $F_M^{(n)} \sim q_\psi(F_M)$, and we use reparameterised samples (e.g. Kingma and Welling, 2014) to be able to compute derivatives wrt. $\psi$. The resulting basic algorithm to evaluate and optimise $\mathcal{L}(\theta, \psi)$ is summarised in Alg. 1.

There are three extensions of this algorithm that are typically required for applying GPSSMs in practice (e.g. in Sec. 6.4.2), i) mini-batches, ii) multi-dimensional latent states and iii) control inputs: Many time series are too long to be handled in one batch such that using i) mini-batches helps obtaining a computationally tractable algorithm. Moreover, for many real-world problems a one-dimensional latent state is not expressive enough (Frigola, 2015) and we require ii) multi-dimensional latent states $x_t$. Lastly, many datasets come with an additional time series $u_t \in \mathbb{R}^{d_u}$ of iii) external inputs that additionally control the behaviour of the system. We briefly summarise these extensions below, for additional details see Lindinger et al. (2022, Appx. E).

Starting with i) we can exploit that the first term in our optimisation objective [Eq. (6.20)] can be written as a sum over the observations $y_t$ [while not directly obvious from Eq. (6.20), this follows from Eqs. (6.10) and (6.7)]. We can approximate the sum using mini-batches (or rather subsequences) of length $T_b$, $Y_{T_b} = \{y_t\}_{t=t_0}^{t_0+T_b}$ starting at some (random) time index $t_0$. Note that this is an approximation that ignores the temporal structure of the data and ignores effects coming from observations and transitions before and after the mini-batch which is discussed in more detail in the previous chapter and in Aicher et al. (2019). Nevertheless, mini-batching nicely integrates with the sampling step in Eq. (6.20), where we can draw a new mini-batch $Y_{T_b}^{(n)}$ for every sample $F_M^{(n)}$ from $q_\psi(F_M)$, resulting in a new approximation,

$$\int q_\psi(F_M) \log \widetilde{p}_\theta(Y_T|F_M) dF_M \approx \frac{T}{T_b} \sum_{n=1}^{N} \log \widetilde{p}_\theta(Y_{T_b}^{(n)}|F_M^{(n)}), \quad F_M^{(n)} \sim q_\psi(F_M).$$

Next, the multi-dimensional latent states $x_t$ needed for ii) are problematic in the transition model in Eq. (2.28), where the mean and the covariance of the GP appear, both of which are one-dimensional. We follow the literature (e.g. Doerr et al., 2018; Ialongo et al., 2019) and choose independent (sparse) GPs for each dimension of the latent state resulting in

$$p_\theta(x_t|x_{t-1}, F_M^{d_x}) = \prod_{d=1}^{d_x} \mathcal{N}\left(x_t^{(d)} \middle| x_{t-1}^{(d)} + \mu^{(d)}(x_{t-1}, F_M^{(d)}), \sigma_x^{(d)} + \Sigma^{(d)}(x_{t-1})\right),$$

where $F_M^{d_x} = \{F_M^{(d)}\}_{d=1}^{d_x}$ is the collection of all inducing outputs, $x_t^{(d)}$ is the $d$-th dimension of the latent state, $\sigma_x^{(d)}$ is the (trainable) transition noise per dimension, and $\mu^{(d)}$ and $\Sigma^{(d)}$ are the mean and covariance of the sparse GP responsible for the $d$-th dimension, respectively [cf. Eq. (2.8)].

Finally, the control inputs for extension iii) are provided as an additional time series $U_T = \{u_t\}_{t=1}^{T}$, where $u_t \in \mathbb{R}^{d_u}$ is applied at time index $t$ and can change the behaviour of the system. In GPSSMs these are typically modelled as additional (constant) inputs to the GP which are simply concatenated with the latent states $x_t$ at the same time index. This leads only to two small changes in the algorithm: The kernel of the GPs have to accept input pairs coming from $\mathbb{R}^{d_x+d_u}$ and the inducing points $X_M$ also have to lie in that higher dimensional space.

## 6.3 Related Work

Before we present an empirical evaluation of our proposed algorithm, we discuss some related work and especially differences to our approach in detail in the following. There are two lines of work that directly relate to our proposed method: The first is on inference techniques for GPSSMs, the second on optimising model parameters for latent variable models using the Laplace approximation. While we already thoroughly introduced the related work on inference in GPSSMs in Sec. 2.4, we will point out the similarities and differences of these works to our approach in Sec. 6.3.1. Then we compare our approach to other works using the Laplace approximation in Sec. 6.3.2 before pointing out some other recent related works [Sec. 6.3.3].

### 6.3.1 Variational Inference in Gaussian Process State-Space Models

The closely related work on inference in GPSSMs begins with Frigola et al. (2014): They used a purely variational inference approach, assuming independence between the temporal latent states $X_{T_0}$ and the inducing outputs $F_M$, $q(F_M, X_{T_0}) = q(F_M)q(X_{T_0})$. This allowed them to find optimal variational distributions $q(F_M)$ and $q(X_{T_0})$ using variational calculus, where the latter distribution is analytically intractable. This leads, similarly as in our work, to a double-loop algorithm, where in the inner loop the distribution $q(X_{T_0})$ is approximated and in the outer loop $q(F_M)$ has to be obtained [cf. Alg. 1]. Frigola et al. (2014) opt for a particle filtering method in the inner loop (note that the Laplace approximation would have also been possible here), but in contrast to our work do not take the conditional dependencies between the $F_M$ and the $X_{T_0}$ into account. Note that this independence assumption allows for alternating updates of the parameters of $q(F_M)$ and $q(X_{T_0})$ without having the need to differentiate through the latent states $X_{T_0}$.

Eleftheriadis et al. (2017) improved upon this method (especially wrt. efficiency) by using a doubly stochastic variational inference scheme that allows for the first time, and similarly as our approach, for mini-batches. They opt for a parametric Gaussian distribution $q(F_M)$, a linear Markov Gaussian model for $q(X_{T_0})$ and additionally employ a recognition model to amortise the inference of the many parameters of such an approach. It is worth noting that our Laplace approximation (implicitly) also leads to a linear Markov Gaussian model as an approximation to the posterior $p(X_{T_0}|Y_T, F_M)$. However, our approach respects the conditional dependence of the temporal states on the inducing outputs $F_M$, and does not require any additional parameters to be learned since the means and variances of the Markov Gaussian model are obtained from the mode $\hat{X}_{T_0}$ and the Hessian $H$ [cf. Eq. (6.18)].

Recent variational methods have also incorporated the dependence between the $F_M$ and the $X_{T_0}$ in their approximations by choosing $q(F_M, X_{T_0}) = q(F_M)q(X_{T_0}|F_M)$: First Doerr et al. (2018), who have simply used the prior for $q(X_{T_0}|F_M)$ and then Ialongo et al. (2019), who employ a parametric non-linear Markov Gaussian model for the same term.[8] While being clearly more expressive than the approximation of Doerr et al. (2018), the functional form in Ialongo et al. (2019) also leads to a harder learning problem through the additional introduction of the parameters of the Gauss-Markov model which is taken to be

$$\begin{aligned} q(X_{T_0}|F_M) &= \prod_{t=0}^{T-1} q(x_{t+1}|x_t, F_M), \\ q(x_{t+1}|x_t, F_M) &= \mathcal{N}\left(x_{t+1}\middle| A_t\mu(x_t, F_M) + b_t, S_t + A_t\Sigma(x_t)A_t^\top\right), \end{aligned} \tag{6.21}$$

where $\mu$ and $\Sigma$ are as in Eq. (2.8) and $A_t$, $b_t$, and $S_t$ are variational parameters that have to be

[8] Note that this is a more expressive approximation than our implicit one in Eq. (6.18), which is contained as a special case in the functional form of Ialongo et al.'s (2019) approximation given in Eq. (6.21).

inferred. Additionally, the above approximation for $q(X_{T_0}|F_M)$ only allows to learn the parameters of $q(x_t|x_{t-1}, F_M)$ after the preceding state $x_{t-1}$ has reached a meaningful state due to the sequential nature of the algorithm. In contrast to these two methods, we do not employ a variational distribution $q(X_{T_0}|F_M)$ but rather approximately marginalise $X_{T_0}|Y_T, F_M$ through the Laplace approximation, eliminating the need to sequentially sample $x_t$ during inference.

### 6.3.2 Laplace Approximation

The other line of related work consists of approaches using the Laplace approximation for parameter estimation in latent variable models. One influential approach is by Rue et al. (2009) that uses the Laplace approximation (twice) to perform approximate Bayesian inference of the model parameters in latent Gaussian models and pays close attention to the sparsity of the Hessian that is induced by different models. The work of Skaug and Fournier (2006) combines the Laplace approximation with automatic differentiation methods to arrive at an algorithm that can be used to approximate the marginal likelihood of (non-) Gaussian latent state models and therefore for maximum likelihood parameter estimation. The software package described in Kristensen et al. (2016) provides a recent implementation of the ideas in Skaug and Fournier (2006) with an additional focus on efficient automatic differentiation exploiting the sparsity of the Hessian. While closely connected to our work, these methods do not cover our approach since they jointly treat all latent variables, $\{F_M, X_{T_0}\}$, which would not lead to a Hessian whose structure can be exploited. Furthermore, a Laplace approximation over all latent variables would also lead to a less expressive approximation: This would entail a joint Gaussianity assumption over $F_M$ and $X_{T_0}$, while our current approach approximates both sets of variables with Gaussian distributions but allows for potentially complex and non-linear interactions.

### 6.3.3 Other Related Work

Casting the net of related work a little wider, we can see a lot of recent interest in the general methods that have also been applied in this work: The Laplace approximation has been used for modern Bayesian neural networks (Ritter et al., 2018; Unlu and Aitchison, 2021), where the latter method also consists of a combination of variational inference and the Laplace approximation, although applied to the same set of parameters. The implicit function theorem has also seen a recent spike in interest as it can be used to differentiate through implicit layers in deep neural networks (see e.g. Gould et al., 2019). The use of both the implicit function theorem and the Laplace approximation (in combination with Hamiltonian Monte Carlo) can be found in Margossian et al., 2020, which is a recent take on the problem of Bayesian parameter estimation in latent Gaussian models (Rue et al., 2009). Furthermore, variational inference remains ever popular as a computationally efficient approximation method for complex probabilistic models such as Bayesian neural networks (Tomczak et al., 2021) or deep Gaussian processes (Lindinger et al., 2020).

## 6.4 Experiments

In this section we validate our newly introduced approach from Sec. 6.2 experimentally. First, in Sec. 6.4.1, we use the controlled environment of a toy dataset called *kink* to test different features of our proposed approach and of the most recent related work by Ialongo et al. (2019). Afterwards, we assess the performance of our method on a range of real-world benchmark datasets (Sec. 6.4.2). Both experiments confirm that our new approach results in better calibrated predictions when compared with the Variationally Coupled Dynamics and Trajectories (VCDT) method from Ialongo et al. (2019)

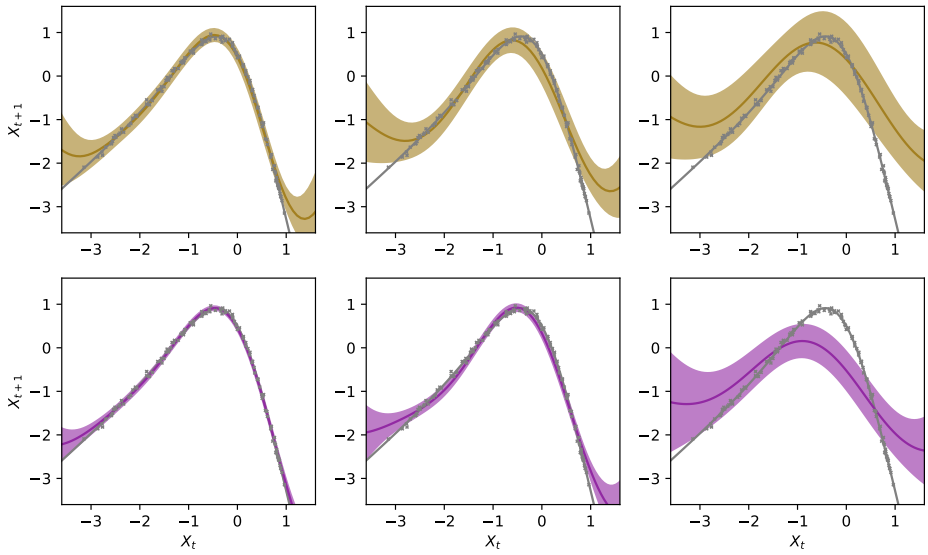


**Figure 6.2:** Sparse GP fits (mean $\pm$ $2\sigma$ confidence interval) to the *kink* transition function for our model (top, brown) and VCDT (bottom, purple) for varying emission noise ($\sigma_y^2 \in \{0.008, 0.08, 0.8\}$, left to right). Each plot additionally shows the *kink* function (solid gray) and one sequence of $T = 120$ noisy latents $x_t$ drawn from that function (tiny gray crosses).

and the Probabilistic Recurrent State-Space Model (PRSSM) method from Doerr et al. (2018). Comparisons to other time-series modelling approaches have already been performed in the latter work in which the PRSSM approach performed best. Hence, we do not repeat these experiments.

### 6.4.1 Kink

The *kink* function $f_k(x) = 0.8 + (x + 0.2)[1 - 5/(1 + e^{-2x})]$ introduced in Ialongo et al. (2019) (see there, Fig. 6.2 or Lindinger et al., 2022, Fig. 5 in Appx. G for visualisations) can be used as a challenging transition function to probe state-space models: It tests the ability to model the non-linear transition function and, by injecting additional noise, also the ability of the inference scheme to deal with different levels of emission noise. We generate data according to

$$x_t \sim \mathcal{N}\left(x_t \middle| f_k(x_{t-1}), \sigma_x^2\right), \quad y_t \sim \mathcal{N}\left(y_t \middle| x_t, \sigma_y^2\right),$$

for $t = 1, \ldots, T$, where we fix $T = 120$, $x_0 = 0.5$, $\sigma_x = 0.05$ and vary $\sigma_y^2 \in \{0.008, 0.08, 0.8\}$. Here, $\sigma_y^2 = 0.8$ corresponds to the setting of Ialongo et al. (2019) for which they empirically demonstrated that the inference scheme of Doerr et al. (2018) is not able to cope with the transition noise $\sigma_x$ and fails to learn the underlying dynamics. We follow Ialongo et al. (2019) and fix the emission model to the groundtruth, $p(y_t|x_t) = \mathcal{N}\left(y_t \middle| x_t, \sigma_y^2\right)$. In addition, we choose a zero-mean sparse GP transition model with trainable Gaussian noise $Q$ [cf. Eq. (2.28)], $p(x_t|x_{t-1}, F_M) = \mathcal{N}\left(x_t|\mu(x_{t-1}, F_M), Q + \Sigma(x_{t-1})\right)$ and a fixed initial distribution $p(x_0) = \mathcal{N}\left(x_0|-0.5, 1.5\right)$, all with one dimensional latent states $x_t$. Further details, the description of the setup of VCDT (Ialongo et al., 2019), and the initialisation and training routines can be found in Lindinger et al. (2022, Appx. E.1).

We present the resulting fits of the sparse GPs to the kink transition function and the noisy latent states in Fig. 6.2. We find that the GP in our model is well able to locate the kink in the *kink* transition function and finds better approximations with increasing signal to noise ratio (decreasing $\sigma_y^2$). The latter is also true for the VCDT method while the former does not hold for the largest $\sigma_y^2 = 0.8$. The

**Table 6.1:** Comparison of our method with VCDT (Ialongo et al., 2019) on the *kink* data set. Shown are mean and standard errors over ten repetitions of the log-density (higher is better) of the *kink* function varying the emission noise variance $\sigma_y^2$. See Lindinger et al. (2022, Appx. E.1) for more details on the execution and evaluation of the experiment.

| Model | $\sigma_y^2 = 0.008$ | $\sigma_y^2 = 0.08$ | $\sigma_y^2 = 0.8$ |
|---|---|---|---|
| Laplace | 1.35(0.04) | **0.36(0.08)** | **-1.08(0.15)** |
| VCDT | 1.53(0.31) | -1.10(0.72) | -4.16(1.97) |

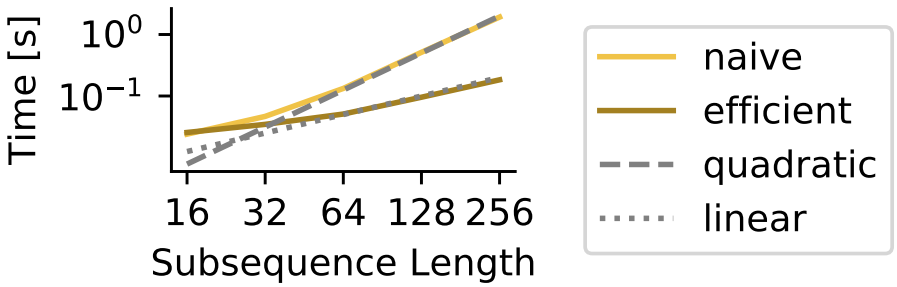


**Figure 6.3:** Timing comparison between a naive Laplace GPSSM and our proposed efficient implementation. The log-log plot shows the average time per training iteration as a function of the subsequence length $T$.

small confidence intervals of the VCDT method sometimes result in very good fits ($\sigma_y^2 = 0.008$), while other times leading to overconfident predictions (slightly for $\sigma_y^2 = 0.08$, evidently for $\sigma_y^2 = 0.8$), an observation that we have also made for the benchmark datasets in Sec. 6.4.2. In contrast, our method provides higher variance estimates resulting in very few data points not lying within the confidence intervals.

Next, we repeat the experiment 10 times, each time generating a new random dataset. Our observations also hold quantitatively as demonstrated in Tab. 6.1. In accordance with Fig. 6.2, we find that VCDT performs slightly, but not significantly, better than our method for $\sigma_y^2 = 0.008$, whereas we significantly outperform VCDT on the noisier data sets.

The controllable environment of the kink dataset also allows us to test if the theoretical speed-ups from Sec. 6.2.3 can be observed in practice. For this, we use the same setup as above, changing only $\sigma_y^2 = 0.01$ to provide an easy learning task and vary $T \in \{2^4, 2^5, 2^6, 2^7, 2^8\}$. We then compare the average runtime per iteration over the first 1000 iterations of our efficient implementation with a naive implementation, where all elements of the Hessian are calculated and the full Hessian is used to compute its determinant and the required Hessian solves, i.e., ignoring the improvements proposed in Sec. 6.2.3. The results of this comparison are depicted in Fig. 6.3 and clearly show that our efficient implementation scales linearly with the length of the time series $T$. For the naive implementation we observe a quadratic scaling with $T$ even though the theoretical scaling is $\mathcal{O}(T^3)$ (see Sec. 6.2.3). We attribute this to the huge cost of calculating the elements of the Hessian with automatic differentiation whose number scales quadratically for the naive version and linearly for our implementation. We hypothesise that the cubic scaling only sets in for very high values of $T$ which might become important if one wants to study long term effects requiring mini-batches of increased size. However, as we saw in Chap. 5, there exist better and more efficient ways to deal with long-term effects instead of simply using long mini-batches. Finally, we also aim to validate our intuition that the parametric approach of Ialongo et al. (2019) for modelling the posterior over the latent states $q(x_t|\cdot)$ [see Eq. 6.21] is problematic from a practical point of view. Their approach requires the

**Table 6.2:** Iteration number at which the mean parameters $A_t$ and $b_t$ of $q(x_t|\cdot)$ [see Eq. (6.21)] of the VCDT method (Ialongo et al., 2019) have converged for different time points $t$. Shown are mean and standard errors over ten repetitions using the *kink* dataset with $T = 120$ and $\sigma_y^2 = 0.08$.

| | $t = 0$ | $t = 40$ | $t = 80$ | $t = 120$ |
|---|---|---|---|---|
| $A_t$ | 4100(600) | 5300(500) | 6200(300) | 6800(400) |
| $b_t$ | 5300(500) | 5500(300) | 6700(400) | 6400(400) |

sequential sampling of $x_t \sim q(x_t|\cdot)$ for $t = 1, \ldots, T$ during training which theoretically means that samples for $x_{t'}$ only become meaningful when the parameters of $q(x_t|\cdot)$ for all $t < t'$ have converged. Our results in Tab. 6.2 support this thesis: There, we show the average iteration number (out of 10000 in total) at which the variational parameters $A_t$ and $b_t$ of the mean of $q(x_t|\cdot)$ have converged when running the VCDT method with the same settings as for Tab. 6.1 on the *kink* data set (see Lindinger et al., 2022, Appx. E.1 for more details). There is a clear trend for $A_t$, and slightly less but still visible for $b_t$, that the variational parameters describing later time points $t$ also converge later during the optimisation. We believe that this small experiment yields a possible explanation for why our approach, even though it uses a theoretically less expressive approximate posterior, outperforms the method of Ialongo et al. (2019).

### 6.4.2 System Identification

We compare the performance of our method against PRSSM (Doerr et al., 2018) and VCDT (Ialongo et al., 2019) on five time series system identification benchmark datasets. Those consist of one dimensional time series of various lengths between 296 and 1024 data points and an equally long time series of one dimensional control inputs (see Fig. 6.4 for visualisations and the appendix of Doerr et al., 2018 for more information about the datasets). For these more complicated tasks we choose a two dimensional latent state $x_t$ and a residual transition model with a sparse GP [cf. Eq. $(2.28)$],

$$p(x_t|x_{t-1}, F_M) = \mathcal{N}\left(x_t|x_{t-1} + \mu(x_{t-1}, F_M), Q + \Sigma(x_{t-1})\right)$$

with (diagonal) trainable Gaussian noise $Q$ and $\mu$ and $\Sigma$ as in Eq. (2.8). We furthermore keep the initial distribution uninformative, $p(x_0) = \mathcal{N}(x_0|0, 1)$, but choose a slightly more expressive emission model (following Ialongo et al., 2019), $p(y_t|x_t) = \mathcal{N}(y_t|Cx_t + b, \Omega)$, where we fix $C = [1, 0]^\top$ and introduce trainable parameters $b$ and $\Omega$. Note that an even more expressive emission model does not lead to more expressivity of the composite model, only to more non-identifiabilities (Frigola, 2015). For the PRSSM and VCDT methods, we use the original models detailed in Doerr et al., 2018 and Ialongo et al., 2019, respectively. For each dataset, we create ten different training tasks by varying the starting index of the training sequence, while keeping the length fixed to one half of the whole time series. Whereas Doerr et al., 2018 only compared the long-term predictions and Ialongo et al., 2019 only the short-term ones, we evaluate both regimes by recording the predictive performance for varying time horizons $T_{\text{test}} \in \{30, 60, 90, 120\}$. For more information about data splits, model configurations as well as training and prediction routines for each method, see Lindinger et al. (2022, Appx. E.2).

We plot the resulting test log-likelihoods in Fig. 6.5 (for a tabular comparison see Lindinger et al., 2022, Tab. 3 in Appx. G). Our results clearly demonstrate that our method is a valuable addition to the set of inference methods for GPSSMs: For short-term predictions ($T_{\text{test}} = 30, 60$), our methods yields excellent results over all datasets, while VCDT shows deteriorated behaviour on *Dryer* and *Gas Furnace*, and PRSSM on *Ballbeam*, *Drive* and *Gas Furnace*. For long-term predictions ($T_{\text{test}} =$

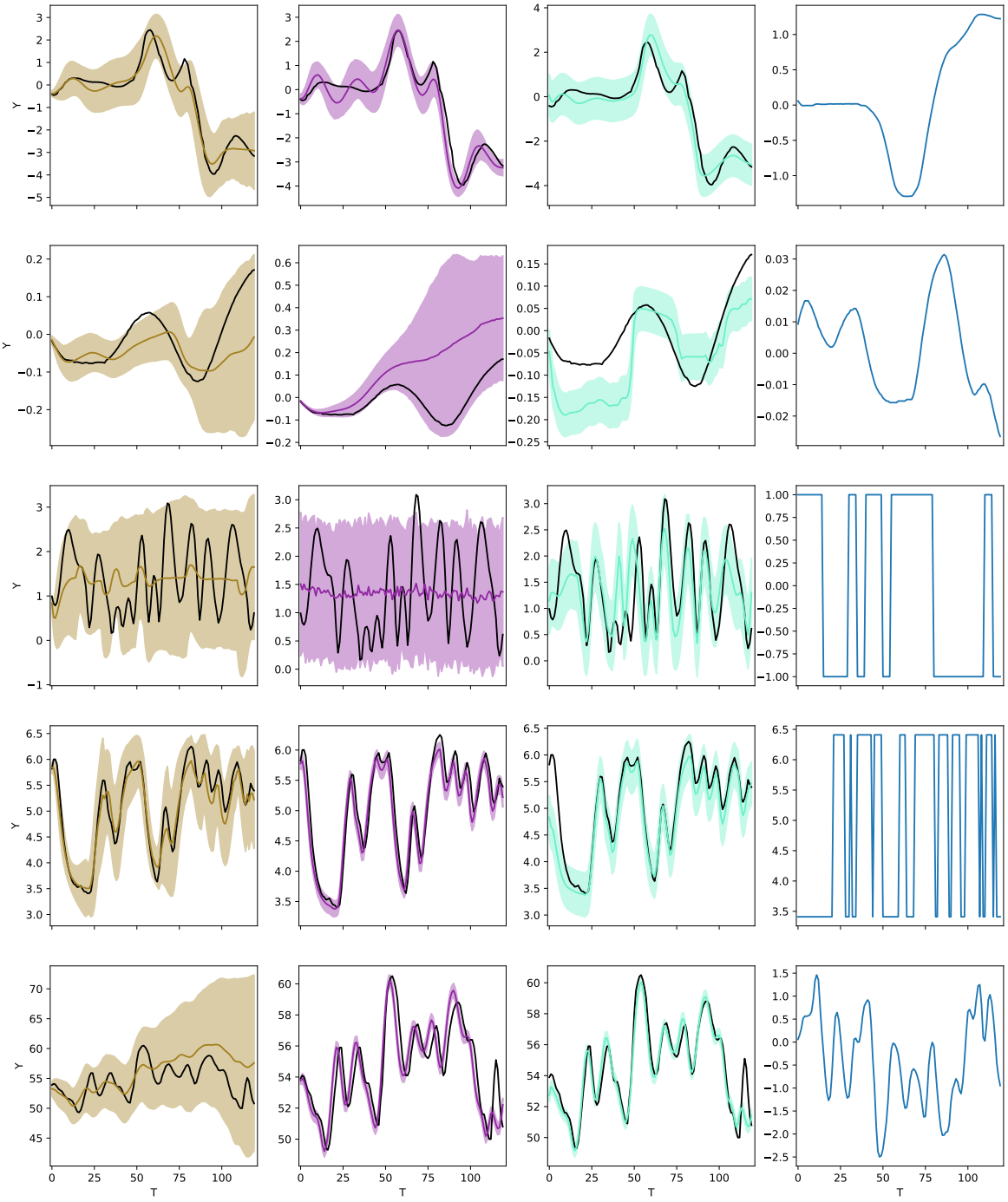


**Figure 6.4:** Exemplary figure showing the differences in predictions for the three different methods on the system identification data sets. Shown are predictions for the same seeds on the *Actuator*, *Ballbeam*, *Drive*, *Dryer*, and *Gas Furnace* datasets in the different rows (top to bottom). From left to right are the predictions for our method, VCDT (Ialongo et al., 2019), PRSSM (Doerr et al., 2018), and the control inputs (all unnormalised).

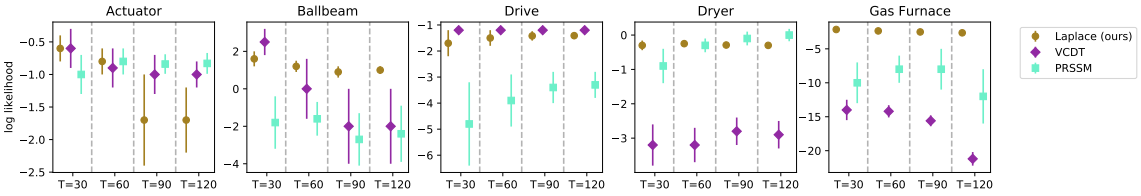


**Figure 6.5:** Comparison of average predictive log-likelihoods (higher is better) and their standard errors for ten repetitions on five different benchmark datasets. We evaluate our model (Laplace), VCDT (Ialongo et al., 2019) and PRSSM (Doerr et al., 2018) on predictions for $T_{\text{test}} \in \{30, 60, 90, 120\}$ steps in the future. See Lindinger et al. (2022, Appx. E.2) for more details about the train-test splits and the evaluation.

$90, 120$), our method significantly outperforms its competitors on *Ballbeam* and *Gas Furnace*, while it underperforms on *Actuator*. From a look at the root-mean-square errors (RMSE) in Lindinger et al. (2022, Tab. 3 in Appx. G), it is evident that a deteriorated log-likelihood value does not necessarily result in a large RMSE-value or vice versa. Instead, a drop in log-likelihood values is more likely caused by overconfident predictions which we can also witness in our exemplary plots in Fig. 6.4.

In sum, we observed in our experiments that our method is able to learn the underlying dynamics for a variety of different tasks and outperforms its comparison partners. A direct attribution of our method's success is unfortunately not possible due to the nature of GPSSMs: They simultaneously learn the outputs of the sparse GP (through the inducing outputs) along with its inputs (the temporal states) such that a distinction between learning the two is impossible as their learning is inherently intertwined. This hinders a better theoretical understanding and also makes it very difficult to clearly attribute predictive improvements to certain parts of the learning process.

However, for a given GP, the Laplace approximation finds an optimal latent state at every iteration which we believe helps the GP convergence. In contrast, the fully variational approximation of Ialongo et al. (2019) performs an incremental update over the latent states in each optimisation step which only leads to an optimal latent state after full convergence. It is therefore possible that the variational approximation of $q(x_t|\cdot)$ for VCDT might be hindered by the optimiser getting stuck in a local optimum. Indeed, our empirical study indicates that VCDT is more susceptible to local optima than our method since the standard errors for VCDT are significantly higher in both experiments. While this provides a plausible explanation, we cannot completely rule out other causes for this behaviour.

Another potential reason for the success of our method is that the chosen variational family of Ialongo et al. (2019) is too compact which can result in narrow uncertainty estimates (e.g. Turner and Sahani, 2011). Our experiments indicate that combining variational inference with the Laplace approximation favours less compact predictive distributions that lead to better calibrated predictions.

## 6.5 Chapter Summary

In this chapter, we have developed a new inference method for Gaussian process state-space models (GPSSMs) that combines a Laplace approximation over the temporal states with variational inference over the inducing outputs of the Gaussian process (GP) part of the model. Our approach learns a joint approximate posterior over the inducing outputs and the temporal states, refraining from sequentially learning the latter. We empirically find that our inference scheme is rewarded by better calibrated predictions compared to state-of-the-art methods.

While we only focused on the application of our inference scheme to GPSSMs, it can generally be applied to all models with two distinct sets of latent variables, e.g. the Bayesian treatment of model parameters in latent Gaussian models (Rue et al., 2009) or of hyperparameters in (sparse) GP models (e.g. Hensman et al., 2013). We further deem exchanging our variational inference engine

with Hamiltonian Monte Carlo (Margossian et al., 2020) as an interesting avenue of future work on GPSSMs since recent research has shown promising results for a fully Bayesian treatment over GP hyperparameters and inducing locations (Rossi et al., 2021).

CHAPTER 7

# Conclusions and Outlook

## 7.1 Summary and Conclusions

In this thesis we introduced new inference methods for deep Gaussian processes (GPs) and Gaussian process state-space models (GPSSMs) based on variational inference after closely investigating the possible trade-offs that previous state-of-the-art methods offered.

Deep GPs and GPSSMs are both specialised probabilistic machine learning methods that embed GPs in a larger model: Deep GPs use a small network of GPs for flexible regression while GPSSMs use a GP as the transition function in a state-space model for time-series modelling. Both models additionally have in common that exact inference is intractable and that recent approaches have mostly relied on variational inference as an approximation method. We exploited these close connections at several points throughout the thesis, e.g. when establishing a common background or by seamlessly transferring the concept of a proof from one model class to the other. Generally, we initially obtained a deep understanding of both model classes and the advantages and disadvantages of existing inference approaches. We then used this understanding in order to design new inference schemes. This was either done by identifying and fixing deficiencies common to all state-of-the-art approaches or consisted of exploring existing trade-offs. In both cases where we did the latter, we traded off the good convergence properties of one method with the expressivity of another one, leading to an empirical improvement over both.

More specifically, we presented the following content in this thesis: In Chap. 2, we first provided detailed technical background on the components that are required for inference in deep GPs and GPSSMs, i.e., GP regression, variational inference, and sparse GPs (an approximation to standard GPs required for their computationally tractable application to large data sets). Building on this, we then thoroughly introduced deep GPs and GPSSMs, including a detailed overview of previous as well as state-of-the-art approaches.

Then, in Chap. 3, we introduced a new variational inference method for deep GPs by generalising an existing state-of-the-art approach, thus making it more expressive. We showed via a proof by induction that a desirable property of the existing approach (the ability to analytically marginalise certain latent variables from the model) is unaffected by our generalisation. This property is known to be advantageous for good convergence properties. A competing state-of-the-art approach does not have this property since it employs an even more expressive approximate inference scheme and therefore has to sample these latent variables. Empirically evaluating our newly proposed method, we found that the analytical marginalisation property indeed improved converge. Additionally, a matrix of (variational) parameters that needs to be optimised for our approach showed an interesting approximate sparsity pattern. After taking this sparsity in our variational inference scheme into account, thus proposing a structured approach, our method proved only to be slightly less computationally efficient than the method it generalises. We empirically assessed that the proposed approach performed comparably to the two state-of-the-art approaches on standard regression benchmarks. Furthermore, we found that it outperformed existing methods when the test data for regression is distant from the training data due to better calibrated predictive uncertainties.

In Chap. 4 we turned our attention to GPSSMs and began to analyse the differences between the

existing state-of-the-art approaches: Both of them use variational inference and approximately marginalise certain latent variables related to the sparse GP in the model but the approximations differ. We proposed a third possibility that allows for exact analytical marginalisation by exploiting the similarities to deep GPs that allowed us to transfer the proof by induction from Chap. 3 to this new setting. Furthermore, we analysed the different sparse GP approaches used by the two methods and were able to conclude that those effectively do not change the model (or rather the optimisation thereof). Finally, we left a major difference, the choice of approximate posteriors used for variational inference, for further scrutiny to Chap. 6.

Next, in Chap. 5, we studied an under-explored problem common to all GPSSM approaches, namely the ability to efficiently (or at all) model time-series consisting of components with different time scales, i.e., slowly varying and quickly changing components. We proposed a new GPSSM model that includes different components each of which has its own time scale. In order to efficiently train this newly proposed model, we established an equivalence between our approach and an approximation to a different time-series modelling paradigm (GP stochastic differential equations). This was possible by strongly relying on the theoretical groundwork laid in the previous chapter. In the experiments we showed that our approach outperforms existing approaches on semi-synthetic data (where multiple components are present) and performs on par with existing approaches on a challenging real-world data set while requiring less fine-tuning.

Finally, in Chap. 6, we picked up the remaining problem from Chap. 4, namely the choice of approximate posteriors used for variational inference in GPSSMs. In similar spirit as in Chap. 3, we proposed a new inference method that trades off the good optimisation properties of one state-of-the-art method with the superior expressivity of the other. We achieved this by combining variational inference with another approximate inference method, the Laplace approximation. After solving several efficiency issues coming from a direct application of the Laplace approximation, we experimentally compared our proposed approach with the two state-of-the-art methods. We found a very good performance of our approach on a toy data set and multiple benchmark data sets due to improved predictive uncertainties.

In conclusion, we provided a thorough and detailed technical introduction to deep GPs and GPSSMs including state-of-the-art methods that can be employed by future researchers to quickly acquaint themselves with the required background in the respective fields. The main contributions of this thesis were one new inference algorithm for deep GPs and GPSSMs, respectively, where we traded off expressivity versus good optimisation properties of existing algorithms, achieving superior performance in the end. These two examples stressed the importance of keeping in mind the problems of non-convex stochastic optimisation: First, for machine learning in general, where it is typically simpler to find a good setting of the model parameters for a model with fewer parameters since this translates into a lower-dimensional optimisation problem. Second, for probabilistic machine learning in particular, where analytical marginalisation instead of sampling (if possible) leads to lower variance estimates which is important for good convergence. The third main contribution was a new efficient inference algorithm for GPSSMs focussing on time series comprised of slowly varying and quickly changing components. This approach generally requires less fine tuning in contrast to other approaches and outperforms those in the scenario it was designed for. Devising this method in an efficient manner was only possible by drawing connections between related but not obviously close fields. This brings us to the possibly obvious final conclusion that a solid (theoretical) understanding of related fields is certainly helpful, if not necessary, to advance ones own field.

## 7.2 Outlook

In the following final section of this thesis we provide a broader overview of the research fields that deep GPs and GPSSMs stem from, namely expressive probabilistic regression and probabilistic time-

series modelling. Our goal is to point out work being done in related fields in order to help identifying under-explored or missing research directions. Additionally, we include concurrent or later works in our overview that complement the contents of this thesis.

### 7.2.1 Deep Gaussian Processes

We start our overview by pointing out some of the concurrent works to our publication on deep GPs (Lindinger et al., 2020) and some approaches that have appeared since then. Most of them provide direct extensions of the publication by Salimbeni and Deisenroth (2017) that our work is based upon, such as allowing for multiple outputs (Kaiser et al., 2018) or multiple input sources (Hamelijnck et al., 2019) (both of which are orthogonal to improving the inference scheme). The approach by Salimbeni et al. (2019) proposes the additional introduction of noisy latent variables that improve predictions especially when the distribution of the outputs is non-Gaussian (e.g. multi-modal).

An interesting view point on deep GPs is taken by Hegde et al. (2019) that interpret them as discretisations of stochastic differential equations. Doing so leads in practice to a deep GP that is significantly deeper than other typical implementations which, however, applies always the same transformation at each layer. A contrasting approach to the standard variational inference schemes is taken by Yu et al. (2019) who interpret inference in deep GPs as a two-player game inspired by generative adversarial networks (Goodfellow et al., 2014) and thus devise a new approach to approximate the posterior.

Several follow-up works considered the treatment of inducing inputs and inducing outputs in deep GPs more closely: Ustyuzhaninov et al. (2020b) and Ober and Aitchison (2021) had the same idea, namely treating the inducing outputs of one layer as the inducing inputs of the next layer (instead of using independent inducing inputs for every layer as proposed by Salimbeni and Deisenroth, 2017). Their approaches only differ in the variational posterior. In another work, Rossi et al. (2021) propose to treat the inducing inputs (and also the GP hyperparameters) in a Bayesian manner, i.e., to assign a prior to them and then to approximate their posterior. This can be seen as an extension of the similar idea of Hensman et al. (2015) for sparse Gaussian processes.

Similarly as in the last example, several new deep GP approaches are based on replacing the variational sparse GP by Hensman et al. (2013) that the deep GP of Salimbeni and Deisenroth (2017) is based on, by a different sparse GP approach. Examples for this are the inter-domain deep GP (Rudner et al., 2020) that extends inter-domain variational Fourier features (Hensman et al., 2017) to deep GPs, or the doubly sparse variational deep GP (Adam et al., 2020) that first extends the state-space formulation of GPs (e.g. Särkkä et al., 2013) to sparse GPs and then to deep GPs. Another such example are convolutional deep GPs (Blomqvist et al., 2019; Dutordoir et al., 2020) or variants thereof (Popescu et al., 2022) that allow deep GPs to scale to higher input dimensions (e.g. images) and are extensions of convolutional sparse GPs (van der Wilk et al., 2017).

As a possible direction for future work, it would be highly interesting to see whether a combination of these approaches with our work is possible and advantageous. Furthermore, we believe that the evaluation of regression models on test data that is far away from the training data (as we did in Lindinger et al., 2020) is important to understand the advantages but also limitations of probabilistic models (see also Popescu et al., 2022 who work on the closely related task of out-of-distribution detection with probabilistic models). It is therefore of high practical relevance to assess how the other approaches (especially those that aim to be more Bayesian) fare on this task.

### 7.2.2 Expressive probabilistic regression

The final examples of deep GP approaches in the previous section, i.e., convolutional deep GPs, extend the capability of this model class into a domain that is predominantly governed by deep neural networks (e.g. Goodfellow et al., 2016). If we are additionally interested in probabilistic approaches,

this leads us to the large model class of Bayesian neural networks (BNNs) (e.g. Neal, 1996; Abdar et al., 2021; Jospin et al., 2022). There exists a wide range of possible inference approaches for BNNs (see e.g. Abdar et al., 2021; Jospin et al., 2022), however many of the ideas that we used throughout this thesis have also been applied to BNNs. This includes the aim to marginalise latent variables (e.g. Tomczak et al., 2021) or to include structure for efficiency (e.g Tomczak et al., 2020; Swiatkowski et al., 2020) which we employed in Lindinger et al. (2020), or using the Laplace approximation (e.g. Unlu and Aitchison, 2021) as we did in Lindinger et al. (2022).

Intriguingly, the connection between deep GPs and BNNs runs much deeper: Similarly as (wide) single layer neural networks can be interpreted as GPs (Neal, 1996), recent work (Agrawal et al., 2020; Pleiss and Cunningham, 2021; Dutordoir et al., 2021) suggests that the same holds true for deep neural networks and deep GPs. This connection can exemplarily be exploited for improving the accuracy of deep GPs or, vice versa, the calibration of BNNs (Dutordoir et al., 2021). It is therefore interesting to consider whether this similarity can be further exploited to transfer recent advances from the BNN community to the (deep) GP world. In the following we focus on two recent trends, deep ensembles (and similar approaches) and the cold posterior effect.

One of the most prominent direction for BNNs recently, is to use *ensembles* of deep neural networks which basically amounts to training multiple neural networks at the same time and then to combine their predictions during testing which empirically leads to better calibrated uncertainties (Lakshminarayanan et al., 2017; Wenzel et al., 2020b). A conceptually similar approach by Maddox et al. (2019) is to use a single network, train it with stochastic gradient descent, and then, essentially, to interpret the parameter configurations during the last iterations of training as different networks that form an ensemble. Similar ideas are also applicable to sparse or deep Gaussian processes where we experimentally often observed that the initialisation of hyper- or variational parameters strongly influenced the result of the model optimisation. Hence it would be interesting to see whether such approaches can help overcome the simple predictive distributions that are often observed for the predictions of sparse and deep GPs and criticised and alleviated by e.g. Havasi et al. (2018) and Salimbeni et al. (2019).

A further interesting topic concerns the observation which was recently made by Wenzel et al. (2020a) that so-called *cold posteriors* (which amounts to overcounting the data) improve the predictive performance of BNNs. This finding, which essentially signifies that a non-Bayesian (or at least not completely Bayesian) treatment of BNNs leads to better results, sparked a lively debate in the research community aiming to find an explanation or a fix for this phenomenon (e.g. Wilson and Izmailov, 2020; Abdar et al., 2021; Izmailov et al., 2021; Aitchison, 2021; Fortuin et al., 2022). Initial work on the cold posterior effect for GP models shows that it persists for GP classification but is absent for regression (Adlam et al., 2020). In future work, it would be extremely interesting to clarify whether the cold posterior effect can also be observed in different sparse or deep GP variants.

Finally, what is missing from a practitioner's perspective is a guideline when to use which model. While there exists an extremely extensive review of recent uncertainty quantification methods in deep learning by Abdar et al. (2021), the main focus lies on neural networks and deep GPs are only briefly touched upon. Especially for regression it would be useful to know how well GPs, deep GPs and BNNs perform for different tasks where the data set size and the number of input dimensions is varied. Potential extensions to such a comparison include applications to stationary or non-stationary data and the consideration of different tasks such as classification or potential down-stream tasks, e.g. active learning or Bayesian optimisation.

### 7.2.3 Probabilistic Time-Series Modelling

The second topic that we were concerned with in this thesis is probabilistic time-series modelling with GPSSMs. Here, a recent new method exists which extends the methods that we build on to cope

better with unstable and partially observed systems (Curi et al., 2020). Another approach by Ensinger et al. (2021) introduces the possibility to conserve certain quantities in GP dynamical systems, e.g. the total energy, (cf. Brüdigam et al., 2022) by using higher-order integration methods in order to go from the system state at one time step, $x_t$, to the next, $x_{t+1}$.

Instead of assuming inherently discretised dynamics, several approaches also try to continuously model the dynamics $x(t)$. One example for such a model are ordinary differential equations (ODEs, see e.g. Särkkä, 2013 and references therein) that model infinitesimal changes in the state via $dx = f(x)dt$. In order to estimate an ODE from data, we have to learn the function $f$ which can be done in various ways. Here we are interested in probabilistic approaches. The most closely related approach to GPSSMs is to place a GP prior on $f$ and then to approximate the corresponding posterior which is done e.g. in Heinonen et al. (2018) and Hegde et al. (2022). As discussed in Sec. 7.2.2, it is possible to replace the GP by a BNN in order to model $f$, which is done for ODEs in Yıldız et al. (2019) and Dandekar et al. (2020). The similarity of these works to GPSSMs should easily allow to transfer several of the techniques used in order to make the above-mentioned methods work well in practice. As possible future research directions it would therefore certainly be interesting to see if the structured physics-inspired latent states from Yıldız et al. (2019) can reduce non-identifiabilities in general GPSSM models. Additionally, we believe that the use of variational multiple shooting introduced in Hegde et al. (2022) could alleviate some of the efficiency issues that general GPSSM models have for long time-series which we addressed in Longi et al. (2022).

Another example for continuous probabilistic time-series models are stochastic differential equations (SDEs, see Øksendal, 2003 or also Särkkä and Solin, 2019 for a more recent introduction with a focus on applications). In contrast to ODEs these models directly include stochasticity by adding a so-called diffusion term to the ODE description, $dx = f(x)dt + g(x)dW$, where $W$ is a stochastic process (e.g. the Wiener process, see also Sec. 5.1). The problem here is to estimate $f$ and $g$ directly from time-series data. Due to the inherent stochasticity in the model description, any kind of model for $f$ and $g$ leads to a probabilistic description. Recently, approaches employing neural networks (e.g. Tzen and Raginsky, 2019; Li et al., 2020) are ubiquitous, following the stir that their application to ODE systems created (Chen et al., 2018). However, there are also several approaches using either BNNs (e.g. Look and Kandemir, 2019; Haußmann et al., 2021) or GPs (e.g. Ruttor et al., 2013; Duncker et al., 2019; Jørgensen et al., 2020; Zhao et al., 2020) to model $f$ and/or $g$. We have already exploited the similarity to discretised GP-SDEs (e.g. Ruttor et al., 2013) in order to improve the efficiency of GPSSM for long time-series in Longi et al. (2022). It would therefore only be natural to assess which improvements can be obtained by building on methods that improve on Ruttor et al. (2013), e.g. Zhao et al. (2020) that proposes a more accurate numerical approximation scheme.

Other interesting recent ideas include e.g. the work by Toth and Oberhauser (2020) that uses GPs to directly model time-series data by employing a signature kernel (which takes complete time-series as inputs), the work by Ustyuzhaninov et al. (2020a) that specialises in modelling monotonic time-series, or the approach by Mikheeva et al. (2022) that aims at probabilistically modelling multiple correlated time-series using multi-output GPs.

We see several avenues for future work in this field: The first, similarly as in Sec. 7.2.1, is the combination of the GP-based approaches with recent improvements to sparse GPs such as treating the inducing points in a Bayesian manner (Rossi et al., 2021) or using convolutional GPs (van der Wilk et al., 2017) in order to increase the input dimensionality that can be effectively handled by the GPs.

The second future research direction concerns the need of most GP-based approaches to cheaply sample from the GP posterior in order to achieve efficient algorithms. In our case this corresponds to using the FITC approximation in Lindinger et al. (2022) and Longi et al. (2022) and this problem is more generally discussed in Hewing et al. (2020). However, recently a solution for this problem was proposed, namely *decoupled sampling* that combines two different sparse GP variants in order

to be able to draw efficiently from the GP posterior (Wilson et al., 2020). While some very recent approaches have already employed this approach (Ensinger et al., 2021; Mikheeva et al., 2022; Hegde et al., 2022), we see a great potential in its widespread adoption in the community, e.g. for expressive algorithms that would be too computationally demanding without decoupled sampling.

The final research direction that we propose concerns the multitude of approaches that we put forward in this section. For a practitioner it is currently very hard (if not impossible) to decide which approach to use in which scenario and to evaluate the advantages and disadvantages of one method over the other ones. This is partly due to the fact that a comparison with other approaches is typically only done within a single class of algorithms and that a review and comparison of different approaches for probabilistic time-series modelling is currently missing. We believe that such a review is important to identify under-explored research directions or synergies between the different field. Additionally an experimental comparison that takes into account different areas where time-series modelling can be applied (e.g. short/long train and test trajectories, regularly or irregularly sampled data, low- or high-dimensional observations) would be extremely useful to spark future ideas.

APPENDIX A

# Detailed Derivations

Here we provide more extensive derivations for certain theoretical results throughout the thesis. Formulas that are often required for these derivations are Eqs. (37), (39), and (40) from Toussaint (2011), which give basic identities for multivariate Gaussian distributions and state, respectively,

$$\int \mathcal{N}\left(x|a+Fy,A\right)\mathcal{N}\left(y|b,B\right)dy = \mathcal{N}\left(x\middle|a+Fb,A+FBF^{\top}\right), \tag{A.1}$$

$$\mathcal{N}\left(x|a,A\right)\mathcal{N}\left(y|b+Fx,B\right) = \mathcal{N}\left(\begin{pmatrix}x\\y\end{pmatrix}\middle|\begin{pmatrix}a\\b+Fa\end{pmatrix},\begin{pmatrix}A & A^{\top}F^{\top}\\FA & B+FA^{\top}F^{\top}\end{pmatrix}\right), \tag{A.2}$$

$$\mathcal{N}\left(\begin{pmatrix}x\\y\end{pmatrix}\middle|\begin{pmatrix}a\\b\end{pmatrix},\begin{pmatrix}A & C\\C^{\top} & B\end{pmatrix}\right) = \mathcal{N}\left(x|a,A\right)\mathcal{N}\left(y\middle|b+C^{\top}A^{-1}(x-a),B-C^{\top}A^{-1}C\right). \tag{A.3}$$

## A.1 Gaussian Process Posterior

In the following we provide an instructive way to derive the GP posterior $p(F_N|Y_N)$ in Eq. $(2.2)$ and the GP predictive in Eq. $(2.3)$ starting from Eqs. $(1.1)$, $p(F_N) = \mathcal{N}\left(F_N|0,K_{NN}\right)$, and $(2.2)$, $p(Y_N|F_N) = \mathcal{N}\left(Y_N\middle|F_N,\sigma_y^2 I_N\right)$ (see e.g. Rasmussen and Williams, 2006, for an alternative derivation). The first step is to construct the joint distribution according to Eq. $(\mathrm{A}.2)$:

$$p(F_N,Y_N) = p(F_N)p(Y_N|F_N) = \mathcal{N}\left(\begin{pmatrix}F_N\\Y_N\end{pmatrix}\middle|\begin{pmatrix}0\\0\end{pmatrix},\begin{pmatrix}K_{NN} & K_{NN}\\K_{NN} & K_{NN}+\sigma_y^2 I_N\end{pmatrix}\right).$$

From this, we can derive $p(Y_N,F_N) = p(F_N|Y_N)p(Y_N)$ [Eq. $(\mathrm{A}.3)$], a different factorisation of the joint, from which we can read off the posterior,

$$p(F_N|Y_N) = \mathcal{N}\left(F_N\middle|K_{NN}\left(K_{NN}+\sigma_y^2 I_N\right)^{-1}Y_N, K_{NN}-K_{NN}\left(K_{NN}+\sigma_y^2 I_N\right)^{-1}K_{NN}\right). \tag{A.4}$$

This is Eq. $(2.2)$ in the main text.

In order to derive the GP predictive distribution $p(f_*|Y_N)$ we use the sum rule of probability theory (see e.g. Bishop, 2006, Sec. 1.2),

$$p(f_*|Y_N) = \int p(f_*|F_N)p(F_N|Y_N)dF_N. \tag{A.5}$$

The missing term, $p(f_*|F_N)$, can be obtained by using that under a GP prior the function values for every finite set of input points are distributed according to a multivariate Gaussian distribution. In particular, this also holds for $X_N \cup \{x_*\}$, such that $p(F_N,f_*) = \mathcal{N}\left(F_N,f_*\middle|0,K_{N\cup *,N\cup *}\right)$ [cf. Eq. $(1.1)$].

Written more explicitly,

$$p(F_N, f_*) = \mathcal{N}\left(\begin{pmatrix} F_N \\ f_* \end{pmatrix} \middle| \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} K_{NN} & K_{*N}^\top \\ K_{*N} & k_{**} \end{pmatrix}\right).$$

Factorising this as $p(F_N, f_*) = p(F_N)p(f_*|F_N)$ using again Eq. $(\mathrm{A}.2)$, we can read off

$$p(f_*|F_N) = \mathcal{N}\left(f_* \middle| K_{*N}K_{NN}^{-1}F_N, k_{**} - K_{*N}K_{NN}^{-1}K_{*N}^\top\right).$$

Plugging this together with Eq. $(\mathrm{A}.4)$ into Eq. $(\mathrm{A}.5)$ and solving the resulting integral using Eq. $(\mathrm{A}.1)$ yields

$$p(f_*|Y_N) = \mathcal{N}\left(f_* \middle| K_{*N}\left(K_{NN} + \sigma_y^2 I_N\right)^{-1} Y_N, k_{**} - K_{*N}\left(K_{NN} + \sigma_y^2 I_N\right)^{-1} K_{*N}^\top\right),$$

which is Eq. $(2.3)$ in the main text.

## A.2 Variational Inference and Evidence Lower Bounds

In this section we first give a thorough introduction to variational inference, including the derivation of its optimisation objective, the evidence lower bound (ELBO). Afterwards, we give detailed derivations of various ELBOs appearing throughout the thesis.

Variational inference aims at finding an approximation $q(\cdot)$ to the true posterior $p(\cdot|Y)$, where $\cdot$ is an arbitrary set of latent variables that we wish to perform inference over and $Y$ is a set of observed variables. This is done by first choosing a parametric variational family for the distribution $q_\psi(\cdot)$, e.g. a multivariate Gaussian with parameters $\psi$ corresponding to its mean and covariance, and then finding an optimal setting $\psi^*$ of the parameters such that the approximate posterior $q_{\psi^*}(\cdot)$ is as close as possible to $p(\cdot|Y)$. Here, closeness is measured by the Kullback-Leibler (KL) divergence $\mathrm{KL}[q_\psi(\cdot)||p(\cdot|Y)]$ that is defined as

$$\mathrm{KL}[q(x)||p(x)] = \int q(x) \log \frac{q(x)}{p(x)} dx \tag{A.6}$$

for arbitrary distributions $q$ and $p$ of a variable $x$. Important properties of the KL divergence are the non-negativity, i.e., $\mathrm{KL}[q(x)||p(x)] \geq 0$, and that the KL divergence is zero if and only if the two inputs are the same, i.e., $\mathrm{KL}[q(x)||p(x)] = 0 \iff q(x) = p(x)$. The problem with this approach is that directly optimising the parameters $\psi$ of $q_\psi(\cdot)$ by minimising $\mathrm{KL}[q_\psi(\cdot)||p(\cdot|Y)]$ is not possible since $p(\cdot|Y)$ is assumed to be intractable (either computationally or more generally this could also be due to other reasons, e.g. analytical intractabilities) and needs to be approximated. Fortunately the optimisation of $\psi$ can still be done, although indirectly by using the seemingly unrelated relation for the log-evidence,

$$\log p(Y) = \mathrm{KL}[q_\psi(\cdot)||p(\cdot|Y)] + \int q_\psi(\cdot) \log \frac{p(Y, \cdot)}{q_\psi(\cdot)} d\cdot\,. \tag{A.7}$$

We provide a detailed derivation of this equation in the following, starting with the log-evidence $\log p(Y)$ and a common manipulation

$$\log p(Y) = \int q_\psi(\cdot) \log p(Y) d\cdot, \tag{A.8}$$

where $q_\psi(\cdot)$ is an arbitrary distribution over the latent variables parametrised by $\psi$. This can be done

since generally

$$\int p(x)g(y)dx = g(y)\int p(x)dx = g(y), \tag{A.9}$$

for arbitrary (normalised) distributions $p(x)$ and functions $g(y)$ of a different variable, where we used the normalisation property $\int p(x)dx = 1$ of arbitrary probability distributions in the last step. Multiplying by cleverly chosen ones inside the logarithm in Eq. (A.8) results in

$$\log p(Y) = \int q_\psi(\cdot)\log\frac{p(Y)p(\cdot|Y)q_\psi(\cdot)}{p(\cdot|Y)q_\psi(\cdot)}d\cdot\,.$$

Using standard rules for the logarithm, we can equally write this as

$$\log p(Y) = \int q_\psi(\cdot)\log\frac{q_\psi(\cdot)}{p(\cdot|Y)}d\cdot + \int q_\psi(\cdot)\log\frac{p(Y)p(\cdot|Y)}{q_\psi(\cdot)}d\cdot\,. \tag{A.10}$$

Recognising a KL divergence [cf. Eq. (A.6)] in the first term on the right hand side (RHS) of Eq. (A.10) and the joint distribution $p(Y,\cdot) = p(Y)p(\cdot|Y)$ (using the chain rule of probability theory) in the numerator of the second term, we can equivalently write

$$\log p(Y) = \mathrm{KL}[q_\psi(\cdot)||p(\cdot|Y)] + \int q_\psi(\cdot)\log\frac{p(Y,\cdot)}{q_\psi(\cdot)}d\cdot,$$

which we recognise as Eq. (A.7).

Back to the problem at hand: Using the non-negativity of the KL divergence, we can lower bound Eq. (A.7) yielding

$$\log p(Y) \geq \int q_\psi(\cdot)\log\frac{p(Y,\cdot)}{q_\psi(\cdot)}d\cdot\,. \tag{A.11}$$

The slack of this bound is the KL divergence that we wish to minimise. Hence, by maximising the term on the RHS of Eq. (A.11), the so-called evidence lower bound (ELBO),

$$\mathcal{L}(\psi) \equiv \int q_\psi(\cdot)\log\frac{p(Y,\cdot)}{q_\psi(\cdot)}d\cdot, \tag{A.12}$$

wrt. $\psi$, the KL divergence $\mathrm{KL}[q_\psi(\cdot)||p(\cdot|Y)]$ can be minimised, thus finding the optimal $\psi^*$. Note that Eq. (A.12) no longer depends on the intractable posterior, only on the easily accessible joint $p(Y,\cdot)$. Note also that the bound in Eq. (A.11) is tight since the KL divergence is zero when $q_\psi(\cdot) = p(\cdot|Y)$. However, generally the true posterior is not included in the chosen variational family and the optimal $q_{\psi^*}(\cdot)$ remains an approximation. It is therefore crucial for accurate variational inference to find a good trade-off between a rich variational family $q_\psi(\cdot)$ and a simple optimisation of the variational parameters $\psi$.

### A.2.1 Sparse GP ELBO

The ELBO for the sparse GP $\mathcal{L}_{\mathrm{SGP}}(\psi)$ in Eq. (2.13) can be largely derived by considering only the conditional assumptions in the joint prior $p(Y_N, F_N, F_M) = p(Y_N|F_N)p(F_N|F_M)p(F_M)$ and in the variational posterior $q_\psi(F_N, F_M) = p(F_N|F_M)q_\psi(F_M)$. The only other important assumption is the iid. likelihood $p(Y_N|F_N) = \prod_{n=1}^N p(y_n|f_n)$ [cf. Eq. (2.1)]. Plugging all this in the general formula

for the ELBO [Eq. (A.12)], yields

$$\mathcal{L}_{\text{SGP}}(\psi) = \int p(F_N|F_M) q_\psi(F_M) \log \frac{\left[\prod_{n=1}^N p(y_n|f_n)\right] \cancel{p(F_N|F_M)} p(F_M)}{\cancel{p(F_N|F_M)} q_\psi(F_M)} dF_N dF_M,$$

where we immediately see that the expensive term $p(F_N|F_M)$ [Eq. (2.7)] that contains $K_{NN}$ cancels out inside the logarithm. Splitting the remaining term in two by using standard logarithm rules,

$$\begin{aligned}\mathcal{L}_{\text{SGP}}(\psi) = &\int p(F_N|F_M) q_\psi(F_M) \log \prod_{n=1}^N p(y_n|f_n) dF_N dF_M \\ &+ \int \cancel{p(F_N|F_M)} q_\psi(F_M) \log \frac{p(F_M)}{q_\psi(F_M)} \cancel{dF_N} dF_M, \end{aligned} \tag{A.13}$$

we see that the integrand of the second term is independent of $F_N$ and that therefore, using Eq. (A.9), another term cancels. We therefore recognise the second term on the RHS of Eq. (A.13) as a negative KL divergence [using logarithm rules and Eq. (A.6)]. Simultaneously, we transform the first term by exploiting that the logarithm of a product is the sum of logarithms (where the summation can then be performed outside of the integral):

$$\mathcal{L}_{\text{SGP}}(\psi) = \sum_{n=1}^N \int p(F_N|F_M) q_\psi(F_M) \log p(y_n|f_n) dF_N dF_M - KL\left[q_\psi(F_M)||p(F_M)\right]. \tag{A.14}$$

Taking a closer look at the first term on the RHS of this equation, we realise that every summand is an integral for which the integrand only depends on $f_n$ but not on $F_{N\setminus n}$ which we use to denote all other GP outputs. Making this distinction for the first term on the RHS of Eq. (A.14) obvious, we write

$$\begin{aligned}&\sum_{n=1}^N \int p(F_N|F_M) q_\psi(F_M) \log p(y_n|f_n) dF_N dF_M \\ =&\sum_{n=1}^N \int p(f_n|F_M) \cancel{p(F_{N\setminus n}|F_M, f_n)} q_\psi(F_M) \log p(y_n|f_n) df_n \cancel{dF_{N\setminus n}} dF_M, \end{aligned} \tag{A.15}$$

where once more Eq. (A.9) is applicable. The conditioning $p(F_N|F_M) = p(F_{N\setminus n}|F_M, f_n) p(f_n|F_M)$ could be done using standard Gaussian calculus [Eq. (A.3)] but the complicated term $p(F_{N\setminus n}|F_M, f_n)$ cancels out anyway, while the other term, the marginal $p(f_n|F_M)$ is very simple for multivariate Gaussians: We simply have to read off the $n$-th element of the mean vector and the element at position $(n, n)$ of the covariance matrix of the distribution $p(F_N|F_M)$ in Eq. (2.7) which in this case are simply given by replacing every occurrence of $N$ with an $n$, i.e.,

$$p(f_n|F_M) = \mathcal{N}\left(F_M \middle| K_{nM} K_{MM}^{-1} F_M, k_{nn} - K_{nM} K_{MM}^{-1} K_{nM}^\top\right) \tag{A.16}$$

The remaining term in Eq. (A.15) can be simplified further by reordering the terms and solving the resulting integral over $F_M$:

$$\sum_{n=1}^N \iint \left[p(f_n|F_M) q_\psi(F_M)\right] dF_M \log p(y_n|f_n) df_n = \sum_{n=1}^N \int q(f_n) \log p(y_n|f_n) df_n. \tag{A.17}$$

Here, $q(f_n) = \int p(f_n|F_M)q_\psi(F_M)dF_M$ can be solved using Eq. (A.16), the functional form of the approximate posterior, $q_\psi(F_M) = \mathcal{N}(F_M|\mu_M, S_M)$, and Eq. (A.1) resulting in [cf. Eq. (2.16)]

$$q(f_n) = \mathcal{N}\left(f_n \middle| K_{nM}K_{MM}^{-1}\mu_M, k_{nn} - K_{nM}K_{MM}^{-1}\left(K_{MM} - S_M\right)K_{MM}^{-1}K_{nM}^\top\right).$$

Finally, recognising the integral on the RHS of Eq. (A.17) as an expectation value [Eq. (2.14)], we can combine Eqs. (A.14) - (A.17) yielding

$$\mathcal{L}_{\text{SGP}}(\psi) = \sum_{n=1}^{N} \mathbb{E}_{q(f_n)}\left[\log p(y_n|f_n)\right] - \text{KL}[q_\psi(F_M)||p(F_M)],$$

which is Eq. (2.13) in the main text.

### A.2.2 Deep GP ELBO

The ELBO for the deep GP $\mathcal{L}_{\text{DGP}}$ in Eq. (2.20) can be largely derived by considering only the conditional assumptions in the joint prior $p(Y_N, F_N, F_M)$ [Eq. (2.18)] and in the variational posterior $q(F_N, F_M)$ [Eq. (2.19)]. The only other important assumption is the iid. likelihood $p(Y_N|F_N^L) = \prod_{n=1}^N p(y_n|f_n^L)$ [Eq. (2.17)]. Plugging everything in the general formula for the ELBO [Eq. (A.12)] yields

$$\begin{aligned}
\mathcal{L}_{\text{DGP}} &= \int q(F_N, F_M) \log \frac{p(Y_N|F_N^L)\cancel{\prod_{l=1}^L p(F_N^l|F_M^l, F_N^{l-1})}\prod_{l=1}^L p(F_M^l)}{q(F_M)\cancel{\prod_{l=1}^L p(F_N^l|F_M^l, F_N^{l-1})}} dF_N dF_M \\
&= \int q(F_N, F_M) \log p(Y_N|F_N^L) dF_N dF_M + \int q(\cancel{F_N}, F_M) \log \frac{\prod_{l=1}^L p(F_M^l)}{q(F_M)} \cancel{dF_N} dF_M \\
&= \int q(F_N, F_M) \log \prod_{n=1}^N p(y_n|f_n^L) dF_N dF_M + \int q(F_M) \log \frac{\prod_{l=1}^L p(F_M^l)}{q(F_M)} dF_M \\
&= \sum_{n=1}^N \int q(f_n^L) \log p(y_n|f_n^L) df_n^L - \text{KL}[q(F_M)||\prod_{l=1}^L p(F_M^l)],
\end{aligned} \tag{A.18}$$

which is Eq. (2.20). The steps are equivalent to the ones for the derivation of the sparse GP ELBO in Appx. A.2.1 and explained there in great detail. In the last step in Eq. (A.18), we introduced the marginal $q(f_n^L)$ that is given by

$$q(f_n^L) = \int q(F_N, F_M) dF_M \prod_{n' \in N \setminus n} df_{n'}^L \prod_{l=1}^{L-1} dF_N^l. \tag{A.19}$$

While requiring a quite subtle argumentation (see Salimbeni and Deisenroth, 2017, Remark 2; or Lindinger et al., 2020, Appx. D), it can be shown that the marginal of the last layer GP output in Eq. (A.19) depends only on the marginals for the same data point of all previous layers:

$$q(f_n^L) = \int \left[\int q(F_M) \prod_{l=1}^L p(f_n^l|F_M^l, f_n^{l-1}) dF_M\right] df_n^1 \cdots df_n^{L-1}. \tag{A.20}$$

Here, the $p(f_n^l|F_M^l, f_n^{l-1})$ are the marginals of the $p(F_N^l|F_M^l, F_N^{l-1})$ in Eq. (2.18) and are given as [cf. Eq. (2.7)],

$$\begin{aligned} p(f_n^l|F_M^l, f_n^{l-1}) &= \prod_{t=1}^{T_l} p(f_n^{l,t}|F_M^{l,t}, f_n^{l-1}), \\ p(f_n^{l,t}|F_M^{l,t}, f_n^{l-1}) &= \mathcal{N}\left(f_n^{l,t}\middle| K_{nM}^{l,t}\left(K_{MM}^{l,t}\right)^{-1} F_M^{l,t}, k_{nn}^{l,t} - K_{nM}^{l,t}\left(K_{MM}^{l,t}\right)^{-1}\left(K_{nM}^{l,t}\right)^\top\right) \end{aligned} \tag{A.21}$$

with e.g. $K_{nM}^{l,t} = \{k^{l,t}(f_n^{l-1}, x_m^{l,t})\}_{m=1}^M$ and similarly for the other terms.

Using the mean-field variational family in Eq. (2.22), and plugging it in the inner integral of Eq. (A.20) results in

$$\int q_{\text{MF}}(F_M)\prod_{l=1}^{L} p(f_n^l|F_M^l, f_n^{l-1})dF_M = \prod_{l=1}^{L}\prod_{t=1}^{T_l}\int q_\psi(F_M^{l,t})p(f_n^{l,t}|F_M^{l,t}, f_n^{l-1})dF_M^{l,t}. \tag{A.22}$$

Hence we simply have to solve independent integrals, each having the form needed to apply Eq. (A.1). Using $q_\psi(F_M^{l,t}) = \mathcal{N}\left(F_M^{l,t}\middle|\mu_M^{l,t}, S_M^{l,t}\right)$ [Eq. (2.22)] and Eq. (A.21), we have

$$\begin{aligned} q(f_n^l|f_n^{l-1}) &\equiv \prod_{t=1}^{T_l}\int q_\psi(F_M^{l,t})p(f_n^{l,t}|F_M^{l,t}, f_n^{l-1})dF_M^{l,t} = \prod_{t=1}^{T_l}\mathcal{N}\left(f_n^{l,t}\middle|\mu_n^{l,t}, \Sigma_n^{l,t}\right), \\ \Sigma_n^{l,t} &= k_{nn}^{l,t} - K_{nM}^{l,t}\left(K_{MM}^{l,t}\right)^{-1}\left(K_{MM}^{l,t} - S_M^{l,t}\right)\left(K_{MM}^{l,t}\right)^{-1}\left(K_{nM}^{l,t}\right)^\top. \end{aligned} \tag{A.23}$$

and $\mu_n^{l,t} = K_{nM}^{l,t}\left(K_{MM}^{l,t}\right)^{-1}\mu_M^{l,t}$. The means and covariances are essentially the same as for the sparse GPs in Eq. (2.16). Taken together, Eqs. (A.20) - (A.23) yield

$$q(f_n^L) = \int\prod_{l=1}^{L} q(f_n^l|f_n^{l-1})df_n^1\cdots df_n^{L-1}, \quad q(f_n^l|f_n^{l-1}) = \prod_{t=1}^{T_l}\mathcal{N}\left(f_n^{l,t}\middle|\mu_n^{l,t}, \Sigma_n^{l,t}\right),$$

which is Eq. (2.23) in the main text.

### A.2.3 GPSSM FITC ELBO

Before we derive the ELBO of the GPSSM model with the FITC approximation used by Doerr et al. (2018), we first show how applying the FITC approximation [Eq. (2.10)] to the general GPSSM prior model in Eq. (2.26) leads to Eqs. (2.27) and (2.28).

Applying the FITC approximation from Eq. (2.10) to the GPSSM model amounts to the approximation

$$\begin{aligned} p(F_T|X_T, F_M) &= \prod_{t=0}^{T-1} p(f_t|f_{0:t-1}, x_{0:t}, F_M) \approx \prod_{t=0}^{T-1} p(f_t|x_t, F_M), \\ p(f_t|x_t, F_M) &= \mathcal{N}\left(f_t\middle| K_{tM}K_{MM}^{-1}F_M, k_{tt} - K_{tM}K_{MM}^{-1}K_{tM}^\top\right), \end{aligned} \tag{A.24}$$

where $p(f_t|x_t, F_M)$ is the typical conditional GP term coming from Eq. (2.10). Plugging this approx-

imation in Eq. (2.26) leads to

$$p_{\text{FITC}}(Y_T, X_{T_0}, F_T, F_M) = p(x_0)p(F_M)\prod_{t=1}^{T} p(y_t|x_t)p(x_t|f_{t-1}, x_{t-1})p(f_{t-1}|x_{t-1}, F_M). \quad \text{(A.25)}$$

Here, we see that the complicated dependencies on the $F_T$ have vanished and that therefore a direct marginalisation of the latter from the model in Eq. (A.25) is possible:

$$\begin{aligned} p(Y_T, X_{T_0}, F_M) &= \int p_{\text{FITC}}(Y_T, X_{T_0}, F_T, F_M) dF_T \\ &= p(x_0)p(F_M)\prod_{t=1}^{T} p(y_t|x_t) \int p(x_t|f_{t-1}, x_{t-1})p(f_{t-1}|x_{t-1}, F_M) df_{t-1} \\ &= p(x_0)p(F_M)\prod_{t=1}^{T} p(y_t|x_t)p(x_t|x_{t-1}, F_M). \end{aligned}$$

This is Eq. (2.27) in the main text. In order to get to the last line, we solved the integrals independently for each $t$. This can be done using the formulas in Eqs. (2.25) and (A.24) for $p(x_t|f_{t-1}, x_{t-1})$ and $p(f_{t-1}|x_{t-1}, F_M)$, respectively, together with Eq. (A.1). This leads to the result reported in Eq. (2.28) for $p(x_t|x_{t-1}, F_M)$.

Next, we derive the ELBO for this model when using the approximate posterior in Eq. (2.29). This derivation is very similar to the one for the deep GP in Sec. A.2.2 [see especially Eq. (A.18)] and we only spell out the most important steps. See also the following section (Appx. A.2.4) for a more detailed ELBO derivation for a GPSSM.

The starting point is plugging in the prior model and the variational family [Eqs. (2.27) and (2.29)] in the general ELBO formula in Eq. (A.12), yielding

$$\begin{aligned} \mathcal{L}_{\text{PRSSM}} &= \int q(X_{T_0}, F_M) \log \frac{p(x_0)p(F_M)\prod_{t=1}^{T} p(y_t|x_t)\cancel{\prod_{t=1}^{T} p(x_t|x_{t-1}, F_M)}}{q(x_0)q(F_M)\cancel{\prod_{t=1}^{T} p(x_t|x_{t-1}, F_M)}} dX_{T_0} dF_M \\ &= \int q(X_{T_0}, F_M) \log \prod_{t=1}^{T} p(y_t|x_t) dX_{T_0} dF_M + \int q(x_0, \cancel{X_T, F_M}) \log \frac{p(x_0)}{q(x_0)} dx_0 \cancel{dX_T dF_M} \\ &\quad + \int q(\cancel{X_{T_0}, F_M}) \log \frac{p(F_M)}{q(F_M)} \cancel{dX_{T_0}} dF_M \\ &= \sum_{t=1}^{T} \int q(x_t) \left[\log p(y_t|x_t)\right] dx_t - \text{KL}\left[q(x_0) \parallel p(x_0)\right] - \text{KL}[q(F_M) \parallel p(F_M)]. \end{aligned} \quad \text{(A.26)}$$

The marginal $q(x_t)$ is given as [with a derivation similarly as in Eq. (A.17)]

$$q(x_t) = \int \left[\prod_{t'=1}^{t} p(x_{t'}|x_{t'-1}, F_M)\right] q(x_0)q(F_M) dF_M dx_{0:t-1}.$$

These are the Eqs. (2.30) and (2.31) appearing in the main text.

### A.2.4 GPSSM VCDT ELBO

Here we provide the derivation of the ELBO of the VCDT method from Ialongo et al. (2019) that cannot be found in as much detail in their original paper (see also Lindinger et al., 2022, Appx. A).

The starting point is plugging the prior model $p(Y_T, X_{T_0}, F_T, F_M)$ in Eq. (2.26) and the variational family $q(X_{T_0}, F_T, F_M)$ in Eq. (2.34) in the general formula for the ELBO [Eq. (A.12)] which yields

$$\mathcal{L}_{\text{VCDT}} = \mathbb{E}_{q(X_{T_0},F_T,F_M)}\left[\sum_{t=1}^{T}\log p(y_t|x_t) + \log\frac{p(x_0)}{q(x_0)} + \log\frac{p(F_M)}{q(F_M)} + \sum_{t=1}^{T}\log\frac{p(x_t|f_{t-1},x_{t-1})}{q(x_t|F_M,x_{t-1})}\right]. \tag{A.27}$$

after noting that the $p(F_T|X_T, F_M)$ terms cancel inside the logarithm and some reordering while using logarithm rules. Employing the definition of the expectation value [Eq. (2.14)], then the property in Eq. (A.9) and then the definition of the KL-divergence [Eq. (A.6)], we identify two KL-terms:

$$\begin{aligned}\mathcal{L}_{\text{VCDT}} = \mathbb{E}_{q(X_{T_0},F_T,F_M)}&\left[\sum_{t=1}^{T}\log p(y_t|x_t) + \sum_{t=1}^{T}\log\frac{p(x_t|f_{t-1},x_{t-1})}{q(x_t|F_M,x_{t-1})}\right]\\ &- \text{KL}\left[q(x_0)\parallel p(x_0)\right] - \text{KL}\left[q(F_M)\parallel p(F_M)\right].\end{aligned} \tag{A.28}$$

In the following we carefully treat the remaining two terms inside the expectation:

$$\begin{aligned}&\mathbb{E}_{q(X_{T_0},F_T,F_M)}\left[\sum_{t=1}^{T}\log p(y_t|x_t)\right]\\ &=\sum_{t=1}^{T}\iiint \cancel{p(F_T|X_T,F_M)}\left[\prod_{t'=1}^{T}q(x_{t'}|F_M,x_{t'-1})\right]q(x_0)q(F_M)\log p(y_t|x_t)dF_M dx_{0:T}\cancel{dF_T}\\ &=\sum_{t=1}^{T}\int\left(\int\left[\prod_{t'=1}^{t}q(x_{t'}|F_M,x_{t'-1})\right]q(x_0)q(F_M)dF_M dx_{0:t-1}\right)\log p(y_t|x_t)dx_t\\ &=\sum_{t=1}^{T}\mathbb{E}_{q(x_t)}\left[\log p(y_t|x_t)\right],\end{aligned} \tag{A.29}$$

where $q(x_t)$ is given by the inner integral [cf. Eq. (2.31)],

$$q(x_t) = \int\left[\prod_{t'=1}^{t}q(x_{t'}|F_M,x_{t'-1})\right]q(x_0)q(F_M)dF_M dx_{0:t-1}. \tag{A.30}$$

In order to derive Eq. (A.29) above, we used the definition of the expectation value [Eq. (2.14)] in the first step and then Eq. (A.9) to get rid of the integrals over the $F_T$ and over all $x_{t'}$ for $t' > t$ in the

second step. The remaining term inside the expectation value in Eq. (A.28) can be treated as follows:

$$
\begin{aligned}
&\mathbb{E}_{q(X_{T_0},F_T,F_M)}\left[\sum_{t=1}^{T}\log\frac{p(x_t|f_{t-1},x_{t-1})}{q(x_t|F_M,x_{t-1})}\right] \\
&=\sum_{t=1}^{T}\int q(x_0)q(F_M)p(F_T|X_T,F_M)\left(\prod_{t'=1}^{T}q(x_{t'}|F_M,x_{t'-1})\right) \\
&\qquad\times\log\frac{p(x_t|f_{t-1},x_{t-1})}{q(x_t|F_M,x_{t-1})}dF_M dx_{0:T}df_{0:T-1} \\
&=\sum_{t=1}^{T}\int\left[\int q(x_0)q(F_M)p(f_{t-1}|x_{t-1},F_M)\prod_{t'=1}^{t-1}q(x_{t'}|F_M,x_{t'-1})dx_{0:t-2}\right] \\
&\qquad\times\left(\int q(x_t|F_M,x_{t-1})\log\frac{p(x_t|f_{t-1},x_{t-1})}{q(x_t|F_M,x_{t-1})}dx_t\right)dF_M df_{t-1}dx_{t-1} \\
&=-\sum_{t=1}^{T}\mathbb{E}_{q(x_{t-1},f_{t-1},F_M)}\left[\mathrm{KL}\left[q(x_t|F_M,x_{t-1})\parallel p(x_t|f_{t-1},x_{t-1})\right]\right].
\end{aligned}
\tag{A.31}
$$

Here, we have used Eq. (A.9) to get rid of the integrals over all $f_{t'}$ for $t' \neq t-1$ and over all $x_{t'}$ for $t' > t$ in the second step and additionally did some reordering of the terms. In the last step, we have identified a (negative) KL-divergence using Eq. (A.6) and have additionally summarised

$$
q(x_{t-1},f_{t-1},F_M)=\int q(x_0)q(F_M)p(f_{t-1}|x_{t-1},F_M)\prod_{t'=1}^{t-1}q(x_{t'}|F_M,x_{t'-1})dx_{0:t-2}. \tag{A.32}
$$

Taken together, Eqs. (A.28) - (A.32) give the result reported in Eq. (2.36) in the main text.

## A.3 Derivations for Laplace Approximated Gaussian Process State-Space Models

In this section we collect detailed derivations for Chap. 6: First, details about the Laplace approximation in Appx. A.3.1, then a brief introduction to the implicit function theorem in Appx. A.3.2. The final two sections, Appxs. A.3.3 and A.3.4 are concerned with exploiting the structure and sparsity of the Hessian coming from the Laplace approximation.

### A.3.1 The Laplace Approximation Applied to State-Space Models

In this section we provide a more extensive derivation for the Laplace approximation applied to parametric state-space models in Sec. 6.1.1 in our notation (Lindinger et al., 2022). See e.g. MacKay (2003), Skaug and Fournier (2006), and Kristensen et al. (2016) for derivations in other notations or contexts. As explained in Sec. 6.1.1, we use the Laplace approximation to approximate the marginal likelihood $p_\theta(Y_T)$ in Eq. (6.2). In some more detail, we do this with the following steps: We first introduce an exponential and a logarithm that cancel each other out,

$$
p_\theta(Y_T)=\int\exp\left[\log p_\theta(Y_T,X_{T_0})\right]dX_{T_0}. \tag{A.33}
$$

Next, we introduce $g(X_{T_0}, \theta)$ as a short-hand for the log-joint [see Eq. (6.3)] and find its mode $\hat{X}_{T_0}$ [see Eq. (6.4)] wrt. the latent states $X_{T_0}$. Then we proceed to perform a second order Taylor expansion of $g(X_{T_0}, \theta)$ around this mode and plug it in Eq. (A.33) leading to

$$p_\theta(Y_T) \approx \int \exp\left[g(\hat{X}_{T_0}, \theta) - \frac{1}{2}\left(\hat{X}_{T_0} - X_{T_0}\right)^\top H \left(\hat{X}_{T_0} - X_{T_0}\right)\right] dX_{T_0}. \tag{A.34}$$

Here $H$ is the negative Hessian of $g$ as in Eq. (6.6) and there is no first order contribution since we are expanding around a mode where, by definition, the Jacobian vanishes. The first term in the exponential in Eq. (A.34) is constant in $X_{T_0}$ and can be pulled out of the integral (exponential and logarithm cancel):

$$p_\theta(Y_T) \approx p_\theta(Y_T, \hat{X}_{T_0}) \int \exp\left[-\frac{1}{2}\left(\hat{X}_{T_0} - X_{T_0}\right)^\top H \left(\hat{X}_{T_0} - X_{T_0}\right)\right] dX_{T_0}. \tag{A.35}$$

Remaining in the integral, we recognise the exponential of a negative quadratic form, i.e., an unnormalised multivariate Gaussian, which generally has the density (see e.g. MacKay, 2003),

$$\mathcal{N}(x|\mu, \Sigma) = \det(2\pi\Sigma)^{-\frac{1}{2}} \exp\left[-\frac{1}{2}(\mu - x)^\top \Sigma^{-1}(\mu - x)\right], \tag{A.36}$$

with mean $\mu$ and covariance matrix $\Sigma$. We therefore recognise $\hat{X}_{T_0}$ as the mean and $H$ as the inverse of the covariance matrix, i.e., the precision matrix, in Eq. (A.35), leading to

$$p_\theta(Y_T) \approx p_\theta(Y_T, \hat{X}_{T_0}) \int \det(2\pi H^{-1})^{\frac{1}{2}} \mathcal{N}\left(X_{T_0} \middle| \hat{X}_{T_0}, H^{-1}\right) dX_{T_0}. \tag{A.37}$$

This integral is very easy to solve since the factor with the determinant does not depend on $X_{T_0}$, leaving us with an integral over a normalised probability density which, by definition, evaluates to one. Therefore

$$\begin{aligned} p_\theta(Y_T) &\approx p_\theta(Y_T, \hat{X}_{T_0}) \det(2\pi H^{-1})^{\frac{1}{2}} \\ &= \sqrt{2\pi}^{d_x(T+1)} p_\theta(Y_T, \hat{X}_{T_0}) \det(H)^{-\frac{1}{2}}, \end{aligned} \tag{A.38}$$

using standard rules for determinants and taking into account that $H$ is of dimensionality $d_x(T+1) \times d_x(T+1)$. This is the formula that can be found in Eq. (6.5).

### A.3.2 The Implicit Function Theorem

In order to obtain the formula for $\frac{\partial \hat{X}_{T_0}(\theta)}{\partial \theta}$ in Eq. (6.15) we start by defining a function $h$, that is the Jacobian of the function $g_{\text{GP}}$ in Eq. (6.11):

$$h(x, \theta, F_M) = -\frac{\partial \log p_\theta(Y_T, X_{T_0} \mid F_M)}{\partial X_{T_0}}\bigg|_{X_{T_0}=x}. \tag{A.39}$$

By definition, plugging in the mode $\hat{X}_{T_0}$ for the fixed value $\hat{\theta}$ of the parameters $\theta$ that has been used when obtaining $\hat{X}_{T_0}$, yields a vanishing Jacobian, $h(\hat{X}_{T_0}(\hat{\theta}), \hat{\theta}, F_M) = 0$. Here, we made the dependence of the mode on the parameter setting $\hat{\theta}$ explicit. Under mild differentiability assumptions, the IFT, roughly speaking, guarantees that $\hat{X}_{T_0}(\theta)$ is in fact a function of $\theta$ that is *implicitly defined*

through a vanishing Jacobian in the vicinity of $\hat{\theta}$, i.e., through the equation

$$h(\hat{X}_{T_0}(\theta), \theta, F_M) = 0, \tag{A.40}$$

and that is also differentiable at $\hat{\theta}$ and in its vicinity. Therefore, we can calculate the total derivative wrt. $\theta$ on both sides of Eq. (A.40), yielding

$$\frac{\partial h(\hat{X}_{T_0}, \theta, F_M)}{\partial \theta} + \left.\frac{\partial h(x, \theta, F_M)}{\partial x}\right|_{x=\hat{X}_{T_0}} \frac{\partial \hat{X}_{T_0}(\theta)}{\partial \theta} = 0, \tag{A.41}$$

where we used the chain rule. Recognising

$$\left.\frac{\partial h(x, \theta, F_M)}{\partial x}\right|_{x=\hat{X}_{T_0}} = H(\theta, F_M),$$

i.e., the Hessian $H$ [Eq. (6.12)] of the function $g_{\text{GP}}$ in Eq. (6.11), we can solve Eq. (A.41) for the required derivative, yielding

$$\frac{\partial \hat{X}_{T_0}(\theta)}{\partial \theta} = -H^{-1}(\theta, F_M) \frac{\partial h(\hat{X}_{T_0}, \theta, F_M)}{\partial \theta}.$$

This is Eq. (6.15) appearing in Sec. 6.2.2.

### A.3.3 Efficiently obtaining the non-zero blocks of the Hessian

In the following we provide some technical details on how we obtain the blocks of the Hessian in Eq. (6.17): Reverse mode automatic differentiation frameworks such as TensorFlow (Abadi et al., 2016) implement efficient vector-Jacobian products. As we can think of the Hessian as the Jacobian of the Jacobian of the function $g_{\text{GP}}$ in Eq. (6.11), we can naively obtain the $d_x(T+1)$ columns of the Hessian in Eq. (6.17) by considering the vector-Hessian products $e_j^\top H$ with all $d_x(T+1)$ unit vectors $e_j = \{\delta_{jj'}\}_{j'=0}^{d_x T} \in \mathbb{R}^{d_x(T+1)}$, where $\delta_{ij}$ is the Kronecker delta. This would require $\mathcal{O}(T^2 d_x^2)$ storage and computation time. As explained in Sec. 6.2.3, this is wasteful since many unnecessary zeros are being calculated.

Therefore, in order to tackle the problem of obtaining only the non-zero blocks of the Hessian, we propose instead to use only vector-Hessian products with the three block-vectors $\widetilde{e}_0$, $\widetilde{e}_1$, and $\widetilde{e}_2$ defined by $\widetilde{e}_k = \{\delta_{k,k'\%3} \mathbb{I}_{d_x}\}_{k'=0}^{T} \in \mathbb{R}^{d_x(T+1) \times d_x}$. Here $\%$ denotes the modulo operation and $\mathbb{I}_{d_x}$ is the identity matrix of size $d_x \times d_x$. As an example, $\widetilde{e}_0 = (\mathbb{I}_{d_x}, 0, 0, \mathbb{I}_{d_x}, \cdots)^\top$ such that

$$\widetilde{e}_0^\top H = (A_0, B_1, B_3^\top, A_3, B_4, B_6^\top, \cdots)^\top,$$

which can be implemented as $d_x$ vector-Hessian products. Similarly $\widetilde{e}_1^\top H = (B_1^\top, A_1, B_2, \cdots)^\top$ and $\widetilde{e}_2^\top H = (0, B_2^\top, A_2, B_3 \cdots)^\top$ such that we can obtain all non-zero elements of the Hessian with only $3d_x$ vector-Hessian products, reducing the memory and time requirements to $\mathcal{O}(T d_x^2)$. After some reshaping and transposing this provides us with the quantities $\{A_t\}_{t=0}^{T}$ and $\{B_t\}_{t=1}^{T}$ which are needed for the further steps in our sparse algorithm in the following section.

### A.3.4 Efficiently calculating the Hessian determinant and performing Hessian solves

Using the the block matrices $A_t$ and $B_t$ from the previous section, we show in this section how they can be used to efficiently calculate the determinant of the Hessian and performing a Hessian solve needed for Eqs. $(6.10)$ and $(6.15)$, respectively. We follow e.g. Koulaei and Toutounian (2007) in noting that the Hessian $H$ in Eq. $(6.17)$ allows the factorisation

$$H = (\Lambda + B^\top)\Lambda^{-1}(\Lambda + B), \tag{A.42}$$

where $B$ is the strictly upper-triangular part of $H$ (consisting only of the different $B_t$ blocks) and $\Lambda$ is the block diagonal matrix of recursively defined blocks

$$\Lambda_0 = A_0, \qquad \Lambda_t = A_t - B_t^\top \Lambda_{t-1}^{-1} B_t, \quad t = 1, \ldots, T. \tag{A.43}$$

The factorisation in Eq. $(\mathrm{A}.42)$ allows us to calculate the determinant of the Hessian that is required in Eq. $(6.10)$ as

$$\det H = \prod_{t=0}^{T} \det \Lambda_t. \tag{A.44}$$

Here we used the determinant rules $\det(\Lambda + B^\top) = \det(\Lambda + B) = \det \Lambda$ (since $\Lambda$ is block diagonal and $B$ and $B^\top$ are strictly upper and lower triangular, respectively), and $\det(CD) = \det C \det D$ (see also Salkuyeh, 2006).

The Hessian solve required in Eq. $(6.15)$ can be done by exploiting Eq. $(\mathrm{A}.42)$ as well, yielding

$$H^{-1} = (\Lambda + B)^{-1}\Lambda(\Lambda + B^\top)^{-1}. \tag{A.45}$$

Therefore a solve with a block banded lower triangular matrix $(\Lambda + B^\top)$, then a matrix multiplication with a block diagonal matrix $(\Lambda)$ followed by a solve with a block banded upper triangular matrix $(\Lambda + B)$ are equivalent to a solve with $H$. Since all of these operations can be performed in $\mathcal{O}(Td_x^3)$ steps, which is true as well for the calculation of the blocks of $\Lambda$ in Eq. $(\mathrm{A}.43)$ and the determinant in Eq. $(\mathrm{A}.44)$, we have achieved the desired theoretical speed-ups.

We note that further speed-ups are possible, such as considering the inverse subset algorithm to calculate the derivative of the logarithm of the determinant of the Hessian (see e.g. Kristensen et al., 2016; Durrande et al., 2019). Furthermore, several inherently sequential parts of the code could be written in C++ as is done in Durrande et al. (2019).

APPENDIX B

# Proofs

Here we provide extensive proofs for certain results throughout the thesis. A formula that we require multiple times is the block matrix inversion lemma, which states that

$$\begin{pmatrix} A & B \\ C & D \end{pmatrix}^{-1} = \begin{pmatrix} A^{-1} + A^{-1}B\widetilde{D}^{-1}CA^{-1} & -A^{-1}B\widetilde{D}^{-1} \\ -\widetilde{D}^{-1}CA^{-1} & \widetilde{D}^{-1} \end{pmatrix}, \tag{B.1}$$

$$\widetilde{D} = D - CA^{-1}B. \tag{B.2}$$

The second set of formulas that we require is about affine transformations of multivariate Gaussians: Given two Gaussian distributed variables $x$ and $y$ that obey

$$p(x|y) = \mathcal{N}\left(x|a + Fy, A\right), \qquad \text{and} \qquad p(y) = \mathcal{N}\left(y|b, B\right), \tag{B.3}$$

the following formulas hold (see e.g. Schön and Lindsten, 2011, for a proof):

$$p(x) = \mathcal{N}\left(x\middle|a + Fb, A + FBF^\top\right), \tag{B.4}$$

$$p(y|x) = \mathcal{N}\left(y\middle|b + BF^\top\left(A + FBF^\top\right)^{-1}\left[x - (a + Fb)\right], B - BF^\top\left(A + FBF^\top\right)^{-1}FB\right). \tag{B.5}$$

Note that since

$$p(x|y)p(y) = p(x, y) = p(y|x)p(x), \tag{B.6}$$

Eqs. (B.4) and (B.5) are particularly useful if we wish to rewrite the product of two Gaussian densities that are as in Eq. (B.3).

## B.1 Analytical Marginalisation for Deep GPs

The aim of this section is to provide a complete proof for Thm. 3.1. We will do this by starting from the formula for $q(f_n^l)$ that we work out in Eq. (3.1),

$$q_{\text{FC}}(f_n^L) = \int \left[\int q_{\text{FC}}(F_M) \prod_{l=1}^{L} p(f_n^l|F_M^l; f_n^{l-1}) dF_M\right] df_n^1 \cdots df_n^{L-1}. \tag{B.7}$$

Comparing to Eq. (3.3), we see that it remains to be shown that indeed

$$\int q_{\text{FC}}(F_M) \prod_{l=1}^{L} p(f_n^l|F_M^l; f_n^{l-1}) dF_M = \prod_{l=1}^{L} q(f_n^l|f_n^{1:l-1}), \tag{B.8}$$

where the distributions $q$ on the right hand side have the properties described in Eqs. (3.3) - (3.5). The terms appearing on the left hand side are given by $q_{\text{FC}}(F_M) = \mathcal{N}\left(F_M|\mu_M, S_M\right)$, which is

interchangeably also denoted as

$$q(F_M^{1:L}) = \mathcal{N}\left(F_M^{1:L}\middle|\mu_M^{1:L}, S_M^{1:L,1:L}\right) = q\begin{pmatrix}F_M^1\\ \vdots \\ F_M^L\end{pmatrix} = \mathcal{N}\left(\begin{pmatrix}F_M^1\\ \vdots \\ F_M^L\end{pmatrix}\middle|\begin{pmatrix}\mu_M^1\\ \vdots \\ \mu_M^L\end{pmatrix}, \begin{pmatrix}S_M^{11} & \cdots & S_M^{1L}\\ \vdots & \ddots & \vdots \\ S_M^{L1} & \cdots & S_M^{LL}\end{pmatrix}\right), \tag{B.9}$$

and [cf. Eq. (A.21)]

$$p(f_n^l|F_M^l; f_n^{l-1}) = \mathcal{N}\left(f_n^l\middle|\widetilde{\mathcal{K}}_{nM}^l F_M^l, \widetilde{\mathcal{K}}_{nn}^l\right). \tag{B.10}$$

Note the slight distinction that we make by using ; in Eq. (B.10) indicating that $f_n^{l-1}$ appears only as an input to a kernel in the formula. Furthermore, we defined

$$\widetilde{\mathcal{K}}_{nM}^l = \mathcal{K}_{nM}^l\left(\mathcal{K}_{MM}^l\right)^{-1} \tag{B.11}$$

$$\widetilde{\mathcal{K}}_{nn}^l = \mathcal{K}_{nn}^l - \mathcal{K}_{nM}^l\left(\mathcal{K}_{MM}^l\right)^{-1}\mathcal{K}_{Mn}^l. \tag{B.12}$$

In order to show that Eq. (B.8) holds, we will introduce a rather technical lemma in the following and prove it later by induction.

**Lemma B.1.** *Given the definitions in Eqs.* (B.9) *and* (B.10), $\forall l = 1, \ldots, L$ *we have*

$$\begin{aligned}&\int q(F_M^{1:L})\prod_{l'=1}^{L} p(f_n^{l'}|F_M^{l'}; f_n^{l'-1})dF_M^{l'}\\ &= \left[\prod_{l'=1}^{l-1} q(f_n^{l'}|f_n^{1:l'-1})\right]\int q(f_n^l, F_M^{l+1:L}|f_n^{1:l-1})\prod_{l'=l+1}^{L} p(f_n^{l'}|F_M^{l'}; f_n^{l'-1})dF_M^{l'},\end{aligned} \tag{B.13}$$

*where* $q(f_n^{l'}|f_n^{1:l'-1})$ *is as in Eq.* (3.3) *and*

$$q(f_n^l, F_M^{l+1:L}|f_n^{1:l-1}) = \mathcal{N}\left(\begin{pmatrix}f_n^l\\ F_M^{l+1:L}\end{pmatrix}\middle|\begin{pmatrix}\hat{\mu}_n^l\\ {}^l\hat{\mu}_M^{l+1:L}\end{pmatrix}, \begin{pmatrix}\hat{\Sigma}_n^l & {}^l\hat{\Sigma}_{nM}^{l,l+1:L}\\ \left({}^l\hat{\Sigma}_{nM}^{l,l+1:L}\right)^\top & {}^l\hat{\Sigma}_M^{l+1:L,l+1:L}\end{pmatrix}\right). \tag{B.14}$$

*Here* $\hat{\mu}_n^l$ *and* $\hat{\Sigma}_n^l$ *are as in Eqs.* (3.4) *and* (3.5), *respectively, and we defined*

$$\begin{aligned}{}^l\hat{\mu}_M^{l+1:L} &= \mu_M^{l+1:L} + S_M^{l+1:L,1:l-1}\mathrm{diag}(\widetilde{\mathcal{K}}_{Mn}^{1:l-1})\left(\widetilde{S}_n^{1:l-1,1:l-1}\right)^{-1}(f_n^{1:l-1} - \widetilde{\mu}_n^{1:l-1})\\ {}^l\hat{\Sigma}_M^{l+1:L,l+1:L} &= S_M^{l+1:L,l+1:L} - S_M^{l+1:L,1:l-1}\mathrm{diag}(\widetilde{\mathcal{K}}_{Mn}^{1:l-1})\left(\widetilde{S}_n^{1:l-1,1:l-1}\right)^{-1}\mathrm{diag}(\widetilde{\mathcal{K}}_{nM}^{1:l-1})S_M^{1:l-1,l+1:L}\\ {}^l\hat{\Sigma}_{nM}^{l,l+1:L} &= \widetilde{\mathcal{K}}_{nM}^l S_M^{l,l+1:L} - \widetilde{S}_n^{l,1:l-1}\left(\widetilde{S}_n^{1:l-1,1:l-1}\right)^{-1}\mathrm{diag}(\widetilde{\mathcal{K}}_{nM}^{1:l-1})S_M^{1:l-1,l+1:L}.\end{aligned} \tag{B.15}$$

In the equations above we used $\mathrm{diag}(A^{1:l})$ to denote the formation of a block diagonal matrix, where the diagonal blocks are given by $A^1, \ldots, A^l$. Note that while we only need one index to label $\hat{\mu}_n^l$ and $\hat{\Sigma}_n^l$, we need several for the objects defined in Eq. (B.15). Take e.g. ${}^l\hat{\Sigma}_M^{l+1:L,l+1:L}$: The upper left index denotes for which $l$ the formula is valid (which will become important when we do the induction step $l \to l+1$). The upper right indices (try to) capture which terms of $S_M$ are most important for the definition, they have nothing to do with the dimensionality of the objects. (In fact, the matrix ${}^l\hat{\Sigma}_M$ contains $L-l-1 \times L-l-1$ blocks of various sizes $T_l M \times T_{l'} M$.) This makes it

easier later on when we do calculations with these objects. Before we prove Lem. B.1, we will first show how its results can be used to prove Thm. 3.1:

**Proof of Theorem 3.1**. We start from Eq. (B.7) which reads

$$q_{\text{FC}}(f_n^L) = \int \left[ \int q_{\text{FC}}(F_M) \prod_{l=1}^{L} p(f_n^l | F_M^l; f_n^{l-1}) dF_M \right] df_n^1 \cdots df_n^{L-1}. \tag{B.16}$$

Obtaining a formula for the inner integral can be done using Lem. B.1 for the case $l = L$, in which case Eq. (B.13) evaluates to

$$\int q(F_M^1, \ldots, F_M^L) \prod_{l=1}^{L} p(f_n^l | F_M^l; f_n^{l-1}) dF_M^l = \left[ \prod_{l=1}^{L-1} q(f_n^l | f_n^{1:l-1}) \right] q(f_n^L | f_n^{1:L-1})$$

since there is nothing left to integrate over. According to Eqs. (3.3) and (B.14) the distribution $q(f_n^L | f_n^{1:L-1})$ has the form necessary to be written as part of the product and we therefore have

$$\int q(F_M^1, \ldots, F_M^L) \prod_{l=1}^{L} p(f_n^l | F_M^l; f_n^{l-1}) dF_M^l = \prod_{l=1}^{L} q(f_n^l | f_n^1, \ldots, f_n^{l-1}).$$

Plugging this into Eq. (B.16) yields

$$q(f_n^L) = \int \prod_{l=1}^{L} q(f_n^l | f_n^{1:l-1}) df_n^1 \cdots df_n^{L-1}, \tag{B.17}$$

where the distributions $q$ on the right hand side have the properties described in Eqs. (3.3) - (3.5). □

For the following lengthy proof we often omit the conditioning of $p(f_n^l | F_M^l; f_n^{l-1})$ on $f_n^{l-1}$ and interchangeably also write $p(f_n^l | F_M^l)$.

**Proof of Lemma B.1**. We prove the lemma by induction:

**Base case** We need to show that Eq. (B.13) holds for $l = 1$, i.e., that

$$\int q(F_M^1, \ldots, F_M^L) \prod_{l=1}^{L} p(f_n^l | F_M^l) dF_M^l = \int q(f_n^1, F_M^2, \ldots, F_M^L) \prod_{l=2}^{L} p(f_n^l | F_M^l) dF_M^l, \tag{B.18}$$

where $q(f_n^1, F_M^2, \ldots, F_M^L)$ is given according to Eqs. (B.14) and (B.15).

In order to do so, we will perform the following steps:

i) Starting with the LHS of Eq. (B.18), we isolate all terms that depend on $F_M^1$:

$$\int \left[ \int q(F_M^1, \ldots, F_M^L) p(f_n^1 | F_M^1) dF_M^1 \right] \prod_{l=2}^{L} p(f_n^l | F_M^l) dF_M^l.$$

ii) In the previous equation, we only consider the inner integral and condition $q$ on $F_M^1$:

$$\int q(F_M^1, \ldots, F_M^L) p(f_n^1 | F_M^1) dF_M^1 = \int q(F_M^1) q(F_M^2, \ldots, F_M^L | F_M^1) p(f_n^1 | F_M^1) dF_M^1.$$

iii) Next, we obtain the joint distribution of the two terms that are conditioned on $F_M^1$:

$$\int q(F_M^1)q(F_M^2,\dots,F_M^L|F_M^1)p(f_n^1|F_M^1)dF_M^1 = \int q(F_M^1)q(f_n^1,F_M^2,\dots,F_M^L|F_M^1)dF_M^1.$$

iv) Then we evaluate the integral:

$$\int q(F_M^1)q(f_n^1,F_M^2,\dots,F_M^L|F_M^1)dF_M^1 = q(f_n^1,F_M^2,\dots,F_M^L).$$

v) Finally, we check that the resulting distribution is given by Eqs. (B.14) and (B.15). This then proves the equality in Eq. (B.18).

Step ii) is the first one where we actually need to calculate something, namely the conditioning of $q(F_M^1,\dots,F_M^L)$. Using its definition in Eq. (B.9) and performing the conditioning according to Eq. (A.3) yields

$$\begin{aligned} q(F_M^1,\dots,F_M^L) &= q(F_M^1)q(F_M^2,\dots,F_M^L|F_M^1) = \mathcal{N}\left(F_M^1\middle|\mu_M^1,S_M^{11}\right)\times \\ &\mathcal{N}\left(F_M^{2:L}\middle|\mu_M^{2:L}+S_M^{2:L,1}\left(S_M^{11}\right)^{-1}(F_M^1-\mu_M^1),S_M^{2:L,2:L}-S_M^{2:L,1}\left(S_M^{11}\right)^{-1}S_M^{1,2:L}\right). \end{aligned} \tag{B.19}$$

For step iii) we use the formula we just obtained for $q(F_M^2,\dots,F_M^L|F_M^1)$ and additionally the conditional $p(f_n^1|F_M^1)$, which according to Eq. (B.10) is given by $\mathcal{N}\left(f_n^1\middle|\widetilde{\mathcal{K}}_{nM}^1F_M^1,\widetilde{\mathcal{K}}_{nn}^1\right)$, and then proceed to build their joint Gaussian distribution:

$$\begin{aligned} &q(f_n^1,F_M^2,\dots,F_M^L|F_M^1) = p(f_n^1|F_M^1)q(F_M^2,\dots,F_M^L|F_M^1) \\ &= \mathcal{N}\left(\begin{pmatrix}f_n^1\\F_M^{2:L}\end{pmatrix}\middle|\begin{pmatrix}\widetilde{\mathcal{K}}_{nM}^1F_M^1\\ \mu_M^{2:L}+S_M^{2:L,1}\left(S_M^{11}\right)^{-1}(F_M^1-\mu_M^1)\end{pmatrix}, \right. \\ &\qquad\left.\begin{pmatrix}\widetilde{\mathcal{K}}_{nn}^1 & 0\\ 0 & S_M^{2:L,2:L}-S_M^{2:L,1}\left(S_M^{11}\right)^{-1}S_M^{1,2:L}\end{pmatrix}\right). \end{aligned} \tag{B.20}$$

In step iv) we perform the integration using $q(F_M^1)=\mathcal{N}\left(F_M^1\middle|\mu_M^1,S_M^{11}\right)$ from Eq. (B.19) for the marginal and the term above for the joint. Applying Eq. (A.1) yields

$$\begin{aligned} &\int q(f_n^1,F_M^2,\dots,F_M^L|F_M^1)q(F_M^1)dF_M^1 \\ &= \mathcal{N}\left(\begin{pmatrix}f_n^1\\F_M^{2:L}\end{pmatrix}\middle|\begin{pmatrix}\widetilde{\mathcal{K}}_{nM}^1\mu_M^1\\ \mu_M^{2:L}+S_M^{2:L,1}\left(S_M^{11}\right)^{-1}(\mu_M^1-\mu_M^1)\end{pmatrix}, \right. \\ &\qquad\begin{pmatrix}\widetilde{\mathcal{K}}_{nn}^1 & 0\\ 0 & S_M^{2:L,2:L}-S_M^{2:L,1}\left(S_M^{11}\right)^{-1}S_M^{1,2:L}\end{pmatrix} \\ &\qquad\left.+\begin{pmatrix}\widetilde{\mathcal{K}}_{nM}^1\\ S_M^{2:L,1}\left(S_M^{11}\right)^{-1}\end{pmatrix}S_M^{11}\begin{pmatrix}\widetilde{\mathcal{K}}_{nM}^1\\ S_M^{2:L,1}\left(S_M^{11}\right)^{-1}\end{pmatrix}^\top\right) \\ &= \mathcal{N}\left(\begin{pmatrix}f_n^1\\F_M^{2:L}\end{pmatrix}\middle|\begin{pmatrix}\widetilde{\mu}_n^1\\ \mu_M^{2:L}\end{pmatrix},\begin{pmatrix}\widetilde{S}_n^{11} & \widetilde{\mathcal{K}}_{nM}^1S_M^{1,2:L}\\ S_M^{2:L,1}\widetilde{\mathcal{K}}_{Mn}^1 & S_M^{2:L,2:L}\end{pmatrix}\right). \end{aligned} \tag{B.21}$$

In order to arrive at the last line we simplified the terms and used the definitions of $\widetilde{\mu}_n^1$ and $\widetilde{S}_n^{11}$ in Eq. (3.6).

Step v) requires us to evaluate Eq. (B.14) for $l = 1$ resulting in

$$\mathcal{N}\left(\begin{pmatrix} f_n^1 \\ F_M^{2:L} \end{pmatrix} \middle| \begin{pmatrix} \hat{\mu}_n^1 \\ {}^1\hat{\mu}_M^{2:L} \end{pmatrix}, \begin{pmatrix} \hat{\Sigma}_n^1 & {}^1\hat{\Sigma}_{nM}^{1,2:L} \\ \left({}^1\hat{\Sigma}_{nM}^{l,2:L}\right)^\top & {}^1\hat{\Sigma}_M^{2:L,2:L} \end{pmatrix}\right),$$

which is the term $q(f_n^1, F_M^2, \ldots, F_M^L)$ on the RHS of Eq. (B.18). Plugging in the definitions from Eq. (B.15) we can easily verify that this last term indeed agrees with Eq. (B.21). Therefore our statement in Lem. B.1 holds for $l = 1$.

**Inductive step** We assume that Lemma B.1 holds for some $l = 1, \ldots, L-1$ (induction hypothesis) and then need to show that it also holds for $l + 1$. That is, assuming that

$$\int q(F_M^{1:L}) \prod_{l'=1}^{L} p(f_n^{l'}|F_M^{l'}) dF_M^{l'} = \left[\prod_{l'=1}^{l-1} q(f_n^{l'}|f_n^{1:l'-1})\right] \int q(f_n^l, F_M^{l+1:L}|f_n^{1:l-1}) \prod_{l'=l+1}^{L} p(f_n^{l'}|F_M^{l'}) dF_M^{l'}, \tag{B.22}$$

holds for some $l$ with the terms on the RHS given by Eqs. (B.14), and (B.15) we need to show that we can also write the previous equation as

$$\left[\prod_{l'=1}^{l} q(f_n^{l'}|f_n^{1:l'-1})\right] \int q(f_n^{l+1}, F_M^{l+2:L}|f_n^{1:l}) \prod_{l'=l+2}^{L} p(f_n^{l'}|F_M^{l'}) dF_M^{l'}, \tag{B.23}$$

where this time the terms are given by Eqs. (B.14), and (B.15) but with $l \to l + 1$.

The way to show this is very similar to the way we showed the base case, the resulting formulas will only look more complicated and we will need one additional step in the beginning:

o) Assuming that Eq. (B.22) holds for some $l$, we can start immediately with the RHS. The first step is to marginalise $f_n^l$ from the distribution $q$ within the integral and show that the resulting marginal $q(f_n^l|f_n^{1:l-1})$ has the correct form to be part of the product in front of the integral:

$$\left[\prod_{l'=1}^{l-1} q(f_n^{l'}|f_n^{1:l'-1})\right] \int q(f_n^l, F_M^{l+1:L}|f_n^{1:l-1}) \prod_{l'=l+1}^{L} p(f_n^{l'}|F_M^{l'}) dF_M^{l'} \tag{B.24}$$

$$= \left[\prod_{l'=1}^{l-1} q(f_n^{l'}|f_n^{1:l'-1})\right] \int q(f_n^l|f_n^{1:l-1}) q(F_M^{l+1:L}|f_n^{1:l}) \prod_{l'=l+1}^{L} p(f_n^{l'}|F_M^{l'}) dF_M^{l'} \tag{B.25}$$

$$= \left[\prod_{l'=1}^{l} q(f_n^{l'}|f_n^{1:l'-1})\right] \int q(F_M^{l+1:L}|f_n^{1:l}) \prod_{l'=l+1}^{L} p(f_n^{l'}|F_M^{l'}) dF_M^{l'}. \tag{B.26}$$

Having done this, we will have to do the exact same steps as in the base case, which we will repeat below with updated indices.

i) Continuing from Eq. (B.26), we isolate all terms that depend on $F_M^{l+1}$:

$$\left[\prod_{l'=1}^{l} q(f_n^{l'}|f_n^{1:l'-1})\right] \int \left[\int q(F_M^{l+1:L}|f_n^{1:l}) p(f_n^{l+1}|F_M^{l+1}) dF_M^{l+1}\right] \prod_{l'=l+2}^{L} p(f_n^{l'}|F_M^{l'}) dF_M^{l'}.$$

ii) Comparing this to Eq. (B.23), we see that it remains to be shown that the inner integral equals $q(f_n^{l+1}, F_M^{l+2:L}|f_n^{1:l})$ [given by Eqs. (B.14) and (B.15)]. Therefore we only consider the inner integral and therein condition $q$ on $F_M^{l+1}$:

$$\begin{aligned}&\int q(F_M^{l+1:L}|f_n^{1:l})p(f_n^{l+1}|F_M^{l+1})dF_M^{l+1}\\ =&\int q(F_M^{l+1}|f_n^{1:l})q(F_M^{l+2:L}|f_n^{1:l},F_M^{l+1})p(f_n^{l+1}|F_M^{l+1})dF_M^{l+1}.\end{aligned}$$

iii) Next, we obtain the joint distribution of the two terms that are conditioned on $F_M^{l+1}$:

$$\begin{aligned}&\int q(F_M^{l+1}|f_n^{1:l})q(F_M^{l+2:L}|f_n^{1:l},F_M^{l+1})p(f_n^{l+1}|F_M^{l+1})dF_M^{l+1}\\ =&\int q(F_M^{l+1}|f_n^{1:l})q(f_n^{l+1},F_M^{l+2:L}|f_n^{1:l},F_M^{l+1})dF_M^{l+1}.\end{aligned}$$

iv) Then we evaluate the resulting integral:

$$\int q(F_M^{l+1}|f_n^{1:l})q(f_n^{l+1},F_M^{l+2:L}|f_n^{1:l},F_M^{l+1})dF_M^{l+1}=q(f_n^{l+1},F_M^{l+2:L}|f_n^{1:l}).$$

v) Finally, we check that the resulting distribution is given by Eqs. (B.14) and (B.15). This then proves the equality of Eqs. (B.22) and (B.23).

Let us begin with step o): According to Eq. (B.14), we have

$$q(f_n^l,F_M^{l+1:L}|f_n^{1:l-1})=\mathcal{N}\left(\begin{pmatrix}f_n^l\\F_M^{l+1:L}\end{pmatrix}\middle|\begin{pmatrix}\hat{\mu}_n^l\\{}^l\hat{\mu}_M^{l+1:L}\end{pmatrix},\begin{pmatrix}\hat{\Sigma}_n^l & {}^l\hat{\Sigma}_{nM}^{l,l+1:L}\\ \left({}^l\hat{\Sigma}_{nM}^{l,l+1:L}\right)^\top & {}^l\hat{\Sigma}_M^{l+1:L,l+1:L}\end{pmatrix}\right), \quad \text{(B.27)}$$

which we condition on $f_n^l$ using Eq. (A.3) (i.e., going from Eq. (B.24) to Eq. (B.25)):

$$\begin{aligned}q(f_n^l,F_M^{l+1:L}|f_n^{1:l-1})=q(f_n^l|f_n^{1:l-1})q(F_M^{l+1:L}|f_n^{1:l})=\mathcal{N}\left(f_n^l\middle|\hat{\mu}_n^l,\hat{\Sigma}_n^l\right)\times\\ \mathcal{N}\left(F_M^{l+1:L}\middle|\,{}^l\hat{\mu}_M^{l+1:L}+\left({}^l\hat{\Sigma}_{nM}^{l+1:L,l}\right)^\top\left(\hat{\Sigma}_n^l\right)^{-1}(f_n^l-\hat{\mu}_n^l),\right. \quad \text{(B.28)}\\ \left.{}^l\hat{\Sigma}_M^{l+1:L,l+1:L}-\left({}^l\hat{\Sigma}_{nM}^{l+1:L,l}\right)^\top\left(\hat{\Sigma}_n^l\right)^{-1}{}^l\hat{\Sigma}_{nM}^{l,l+1:L}\right)\end{aligned}$$

We therefore see that $q(f_n^l|f_n^{1:l-1})=\mathcal{N}\left(f_n^l\middle|\hat{\mu}_n^l,\hat{\Sigma}_n^l\right)$, which is the right form for it to be included in the product in front of the integral in Eq. (B.25). This lets us arrive at Eq. (B.26), hence finishing step o).

In step i) nothing really happens, we just note that, according to Eq. (B.10),

$$p(f_n^{l+1}|F_M^{l+1})=\mathcal{N}\left(f_n^{l+1}\middle|\widetilde{K}_{nM}^{l+1}F_M^l,\widetilde{K}_{nn}^{l+1}\right). \quad \text{(B.29)}$$

Using $q(F_M^{l+1:L}|f_n^{1:l})$ from Eq. (B.28), we perform step ii) according to Eq. (A.3), resulting in

$$q(F_M^{l+1:L}|f_n^{1:l})=q(F_M^{l+1}|f_n^{1:l})q(F_M^{l+2:L}|f_n^{1:l},F_M^{l+1}),$$

where

$$q(F_M^{l+1}|f_n^{1:l}) = \mathcal{N}\left(F_M^{l+1}\middle|{}^l\hat{\mu}_M^{l+1} + \left({}^l\hat{\Sigma}_{nM}^{l+1,l}\right)^\top \left(\hat{\Sigma}_n^l\right)^{-1}(f_n^l - \hat{\mu}_n^l), \right. \\ \left. {}^l\hat{\Sigma}_M^{l+1,l+1} - \left({}^l\hat{\Sigma}_{nM}^{l+1,l}\right)^\top \left(\hat{\Sigma}_n^l\right)^{-1} {}^l\hat{\Sigma}_{nM}^{l,l+1}\right) \tag{B.30}$$

and

$$\begin{aligned} &q(F_M^{l+2:L}|f_n^{1:l}, F_M^{l+1}) \\ &= \mathcal{N}\left(F_M^{l+2:L}\middle|{}^l\hat{\mu}_M^{l+2:L} + \left({}^l\hat{\Sigma}_{nM}^{l+2:L,l}\right)^\top \left(\hat{\Sigma}_n^l\right)^{-1}(f_n^l - \hat{\mu}_n^l)\right. \\ &\qquad + \sigma_M \widetilde{\sigma}_M^{-1}\left(F_M^{l+1} - {}^l\hat{\mu}_M^{l+1} - \left({}^l\hat{\Sigma}_{nM}^{l+1,l}\right)^\top \left(\hat{\Sigma}_n^l\right)^{-1}(f_n^l - \hat{\mu}_n^l)\right), \\ &\qquad \left. {}^l\hat{\Sigma}_M^{l+2:L,l+2:L} - \left({}^l\hat{\Sigma}_{nM}^{l+2:L,l}\right)^\top \left(\hat{\Sigma}_n^l\right)^{-1} {}^l\hat{\Sigma}_{nM}^{l,l+2:L} - \sigma_M \widetilde{\sigma}_M^{-1}\sigma_M^\top\right). \end{aligned} \tag{B.31}$$

Here, due to space constraints we used the shorthand notations

$$\begin{aligned} \sigma_M &= {}^l\hat{\Sigma}_M^{l+2:L,l+1} - \left({}^l\hat{\Sigma}_{nM}^{l+2:L,l}\right)^\top \left(\hat{\Sigma}_n^l\right)^{-1} {}^l\hat{\Sigma}_{nM}^{l,l+1}, \\ \widetilde{\sigma}_M &= {}^l\hat{\Sigma}_M^{l+1,l+1} - \left({}^l\hat{\Sigma}_{nM}^{l+1,l}\right)^\top \left(\hat{\Sigma}_n^l\right)^{-1} {}^l\hat{\Sigma}_{nM}^{l,l+1}. \end{aligned}$$

For step iii) we have to build the joint Gaussian distribution

$$q(F_M^{l+2:L}|f_n^{1:l}, F_M^{l+1})p(f_n^{l+1}|F_M^{l+1}) = q(f_n^{l+1}, F_M^{l+2:L}|f_n^{1:l}, F_M^{l+1}) \tag{B.32}$$

using Eqs. (B.29) and (B.31). Since this formula would be even longer than the one in Eq. (B.31), we refrain from explicitly writing it here. While the corresponding formula for the base case [Eq. (B.20)] is much simpler the resulting form of Eq. (B.32) would be similar.

Next, the integration in step iv) can be performed using Eqs. (A.1), (B.30), and (B.32). The calculations are again very similar to the ones in the corresponding step for the base case [Eq. (B.21)] so we only state the final result here:

$$\begin{aligned} q(f_n^{l+1}, F_M^{l+2:L}|f_n^{1:l}) &= \int q(F_M^{l+1}|f_n^{1:l}) q(f_n^{l+1}, F_M^{l+2:L}|f_n^{1:l}, F_M^{l+1}) dF_M^{l+1} \\ &= \mathcal{N}\left(\begin{pmatrix} f_n^{l+1} \\ F_M^{l+2:L} \end{pmatrix}\middle| \begin{pmatrix} \hat{m}_n^{l+1} \\ \hat{m}_M^{l+2:L} \end{pmatrix}, \begin{pmatrix} \hat{S}_n^{l+1} & \hat{S}_{nM}^{l+1,l+2:L} \\ \left(\hat{S}_{nM}^{l+1,l+2:L}\right)^\top & \hat{S}_M^{l+2:L,l+2:L} \end{pmatrix}\right), \end{aligned} \tag{B.33}$$

where

$$\hat{m}_n^{l+1} = \widetilde{\mathcal{K}}_{nM}^{l+1}\left({}^{l}\hat{\mu}_M^{l+1} + \left({}^{l}\hat{\Sigma}_{nM}^{l+1,l}\right)^\top \left(\hat{\Sigma}_n^l\right)^{-1}\left(f_n^l - \hat{\mu}_n^l\right)\right) \tag{B.34}$$

$$\hat{m}_M^{l+2:L} = {}^{l}\hat{\mu}_M^{l+2:L} + \left({}^{l}\hat{\Sigma}_{nM}^{l+2:L,l}\right)^\top \left(\hat{\Sigma}_n^l\right)^{-1}\left(f_n^l - \hat{\mu}_n^l\right) \tag{B.35}$$

$$\hat{S}_n^{l+1} = \mathcal{K}_{nn}^{l+1} + \mathcal{K}_{nM}^{l+1}\left({}^{l}\hat{\Sigma}_M^{l+1,l+1} - \left({}^{l}\hat{\Sigma}_{nM}^{l+1,l}\right)^\top \left(\hat{\Sigma}_n^l\right)^{-1} {}^{l}\hat{\Sigma}_{nM}^{l,l+1}\right)\mathcal{K}_{Mn}^{l+1} \tag{B.36}$$

$$\hat{S}_{nM}^{l+1,l+2:L} = \widetilde{\mathcal{K}}_{nM}^{l+1}\left({}^{l}\hat{\Sigma}_M^{l+1,l+2:L} - \left({}^{l}\hat{\Sigma}_{nM}^{l+1,l}\right)^\top \left(\hat{\Sigma}_n^l\right)^{-1} {}^{l}\hat{\Sigma}_{nM}^{l,l+2:L}\right) \tag{B.37}$$

$$\hat{S}_M^{l+2:L,l+2:L} = {}^{l}\hat{\Sigma}_M^{l+2:L,l+2:L} - \left({}^{l}\hat{\Sigma}_{nM}^{l+2:L,l}\right)^\top \left(\hat{\Sigma}_n^l\right)^{-1} {}^{l}\hat{\Sigma}_{nM}^{l,l+2:L}. \tag{B.38}$$

What remains to be shown in step v) is that this result does in fact agree with the expected result from Lem. B.1, i.e.,

$$q(f_n^{l+1}, F_M^{l+2:L} | f_n^{1:l}) = \mathcal{N}\left(\begin{pmatrix} f_n^{l+1} \\ F_M^{l+2:L}\end{pmatrix} \middle| \begin{pmatrix} \hat{\mu}_n^{l+1} \\ {}^{l+1}\hat{\mu}_M^{l+2:L}\end{pmatrix}, \begin{pmatrix} \hat{\Sigma}_n^{l+1} & {}^{l+1}\hat{\Sigma}_{nM}^{l+1,l+2:L} \\ \left({}^{l+1}\hat{\Sigma}_{nM}^{l+1,l+2:L}\right)^\top & {}^{l+1}\hat{\Sigma}_M^{l+2:L,l+2:L}\end{pmatrix}\right), \tag{B.39}$$

where the terms are defined in Eqs. (3.4), (3.5), and (B.15). That means we have to prove that $\hat{m}_n^{l+1} = \hat{\mu}_n^{l+1}$ and similarly for the other terms in Eqs. (B.35) - (B.38). Note that this is the point where we need the left indices in order to distinguish e.g. the term ${}^{l}\hat{\mu}_M^{l+2:L}$ appearing in Eq. (B.35) from ${}^{l+1}\hat{\mu}_M^{l+2:L}$ appearing in the mean of Eq. (B.39).

We will exemplarily prove that $\hat{m}_n^{l+1} = \hat{\mu}_n^{l+1}$: Starting from Eq. (B.34) we have

$$\begin{aligned}
\hat{m}_n^{l+1} &= \widetilde{\mathcal{K}}_{nM}^{l+1}\left({}^{l}\hat{\mu}_M^{l+1} + \left({}^{l}\hat{\Sigma}_{nM}^{l+1,l}\right)^\top \left(\hat{\Sigma}_n^l\right)^{-1}\left(f_n^l - \hat{\mu}_n^l\right)\right) \\
&= \widetilde{\mu}_n^{l+1} + \widetilde{S}_n^{l+1,1:l-1}\left(\widetilde{S}_n^{1:l-1,1:l-1}\right)^{-1}\left(f_n^{1:l-1} - \widetilde{\mu}_n^{1:l-1}\right) \\
&\quad + \left(\widetilde{S}_n^{l+1,l} - \widetilde{S}_n^{l+1,1:l-1}\left(\widetilde{S}_n^{1:l-1,1:l-1}\right)^{-1}\widetilde{S}_n^{1:l-1,l}\right)\times \\
&\quad \left(\hat{\Sigma}_n^l\right)^{-1}\left(f_n^l - \widetilde{\mu}_n^l - \widetilde{S}_n^{l,1:l-1}\left(\widetilde{S}_n^{1:l-1,1:l-1}\right)^{-1}\left(f_n^{1:l-1} - \widetilde{\mu}_n^{1:l-1}\right)\right),
\end{aligned} \tag{B.40}$$

where we used the definitions in Eqs. (3.4) and (B.15) for the terms $\hat{\cdot}$. Note that these definitions are part of the induction hypothesis. It will soon become clear why we did not substitute $\hat{\Sigma}_n^l$. We furthermore used the definitions of the $\widetilde{\mu}_n$ and $\widetilde{S}_n$ terms in Eq. (3.6) to absorb the $\widetilde{\mathcal{K}}$ terms.

In the following we are going to write Eq. (B.40) in a vectorised form and additionally substitute

$$A = \widetilde{S}_n^{1:l-1,1:l-1}, \qquad B = \widetilde{S}_n^{1:l-1,l}, \qquad C = \widetilde{S}_n^{l,1:l-1}, \qquad \widetilde{D} = \hat{\Sigma}_n^l. \tag{B.41}$$

The reason for these steps will become clear after two more equations:

$$\hat{m}_n^{l+1} = \widetilde{\mu}_n^{l+1} + \begin{pmatrix} \widetilde{S}_n^{l+1,1:l-1}A^{-1} - \left(\widetilde{S}_n^{l+1,l} - \widetilde{S}_n^{l+1,1:l-1}A^{-1}B\right)\widetilde{D}^{-1}CA^{-1} \\ \left(\widetilde{S}_n^{l+1,l} - \widetilde{S}_n^{l+1,1:l-1}A^{-1}B\right)\widetilde{D}^{-1}\end{pmatrix}^\top \begin{pmatrix} f_n^{1:l-1} - \widetilde{\mu}_n^{1:l-1} \\ f_n^l - \widetilde{\mu}_n^l\end{pmatrix}. \tag{B.42}$$

Going one step further, we recognise this as a vector matrix multiplication,

$$\hat{m}_n^{l+1} = \widetilde{\mu}_n^{l+1} + \begin{pmatrix} \widetilde{S}_n^{l+1,1:l-1} \\ \widetilde{S}_n^{l+1,l} \end{pmatrix}^\top \begin{pmatrix} A^{-1} + A^{-1}B\widetilde{D}^{-1}CA^{-1} & -A^{-1}B\widetilde{D}^{-1} \\ -\widetilde{D}^{-1}CA^{-1} & \widetilde{D}^{-1} \end{pmatrix} \begin{pmatrix} f_n^{1:l-1} - \widetilde{\mu}_n^{1:l-1} \\ f_n^l - \widetilde{\mu}_n^l \end{pmatrix}, \tag{B.43}$$

where we additionally exploited that $A$ and $\widetilde{D}$ are symmetric and that $B^\top = C$. In order to get any further from here we need the block matrix inversion lemma in Eq. (B.1). Comparing Eqs. (B.43) and (B.1) explains why we insisted on vectorising the last few formulas and also our definitions in Eq. (B.41). Finally, since $\hat{\Sigma}_n^l = \widetilde{S}_n^{ll} - \widetilde{S}_n^{l,1:l-1}\left(\widetilde{S}_n^{1:l-1,1:l-1}\right)^{-1}\widetilde{S}_n^{1:l-1,l}$ [Eq. (3.5)], we also identify $\widetilde{S}_n^{ll} = D$ [cf. Eq. (B.2)]. We can therefore rewrite Eq. (B.43) by reversing the block matrix inversion and resubstituting the terms in Eq. (B.41):

$$\begin{aligned} \hat{m}_n^{l+1} &= \widetilde{\mu}_n^{l+1} + \begin{pmatrix} \widetilde{S}_n^{l+1,1:l-1} \\ \widetilde{S}_n^{l+1,l} \end{pmatrix}^\top \begin{pmatrix} \widetilde{S}_n^{1:l-1,1:l-1} & \widetilde{S}_n^{1:l-1,l} \\ \widetilde{S}_n^{l,1:l-1} & \widetilde{S}_n^{ll} \end{pmatrix}^{-1} \begin{pmatrix} f_n^{1:l-1} - \widetilde{\mu}_n^{1:l-1} \\ f_n^l - \widetilde{\mu}_n^l \end{pmatrix} \\ &= \widetilde{\mu}_n^{l+1} + \widetilde{S}_n^{l+1,1:l}\left(\widetilde{S}_n^{1:l,1:l}\right)^{-1}\left(f_n^{1:l} - \widetilde{\mu}_n^{1:l}\right). \end{aligned} \tag{B.44}$$

In the last step we simply rewrote the vectors and the matrix according to the way we defined the submatrix notation. Comparing the final result to Eq. (3.4), we realise that this is indeed $\hat{\mu}_n^{l+1}$, i.e., the mean term where we substituted $l \to l+1$. In exactly the same way, i.e., by reversing the matrix inversion, we can show that the other parameters of the distribution in Eq. (B.33) indeed coincide with the respective parameters of the distribution in Eq. (B.39). Since this was the last part that remained to be shown, we finished the proof of Lem. B.1. □

## B.2 Analytical Marginalisation for GPSSMs

The aim of this section is to provide a complete proof for Thm. 4.1. We will do this by starting from the slightly reformulated formula for $q(x_t)$ in Eq. (4.1),

$$q(x_t) = \int q(x_0)\left[\int q(F_M, x_t, \ldots, x_1|x_0)dF_M\right]dx_{0:t-1}, \tag{B.45}$$

where the individual terms are given by (repeated here for convenience),

$$q(F_M, x_t, \ldots, x_1|x_0) = q(F_M)\prod_{t'=0}^{t-1} p(x_{t'+1}|x_{t'}, F_M), \tag{B.46}$$

$$q(x_0) = \mathcal{N}(x_0|m_0, S_0), \tag{B.47}$$

$$q(F_M) = \mathcal{N}(F_M|m_M, S_M), \tag{B.48}$$

$$p(x_{t+1}|x_t, F_M) = \mathcal{N}\left(x_{t+1}\middle|K_{tM}K_{MM}^{-1}F_M, \sigma_x^2 + k_{tt} - K_{tM}K_{MM}^{-1}K_{tM}^\top\right). \tag{B.49}$$

Comparing to Eq. (4.2), we see that we have to show that

$$\int q(x_0)\left[\int q(F_M, x_t, \ldots, x_1|x_0)dF_M\right]dx_{0:t-1} = \int q(x_0)\left[\prod_{t'=1}^{t} q(x_{t'}|x_{t'-1}, \ldots, x_0)\right]dx_{0:t-1}, \tag{B.50}$$

where the distributions $q(x_{t'}|x_{t'-1},\dots,x_0)$ on the right hand side have the properties described in Eqs. (4.3) – (4.5). In order to prove this equality and therefore Thm. 4.1, we require the following technical lemma (Longi et al., 2021):

**Lemma B.2.** *The term $q(F_M, x_t, \dots, x_1|x_0)$ in Eq. (B.46) can also be written as*

$$q(F_M, x_t, \dots, x_1|x_0) = q(F_M|x_t, \dots, x_0) \prod_{t'=1}^{t} q(x_{t'}|x_{t'-1}, \dots, x_0), \tag{B.51}$$

*for $t \in \{1, \dots, T\}$. Here, the $q(x_t|x_{t-1}, \dots, x_0)$ are as in Eqs. (4.3) -(4.5) and*

$$q(F_M|x_t, \dots, x_0) = \mathcal{N}\left(F_M \middle| \hat{\mu}_M^t, \hat{\Sigma}_M^t\right), \tag{B.52}$$

$$\hat{\mu}_M^t = m_M + S_M K_{MM}^{-1} \left(K_{0:t-1,M}\right)^\top \widetilde{S}_{0:t-1,0:t-1}^{-1} \left(x_{1:t} - \widetilde{\mu}_{0:t-1}\right), \tag{B.53}$$

$$\hat{\Sigma}_M^t = S_M - S_M K_{MM}^{-1} \left(K_{0:t-1,M}\right)^\top \widetilde{S}_{0:t-1,0:t-1}^{-1} K_{0:t-1,M} K_{MM}^{-1} S_M. \tag{B.54}$$

With a slight abuse of the slicing notation, we denote $\left(K_{0:t-1,M}\right)^\top = \begin{pmatrix} K_{0,M}^\top & \cdots & K_{t-1,M}^\top \end{pmatrix} \in \mathbb{R}^{M \times t}$. Before providing the proof of Lem. B.2, we show how this lemma can be used to prove Thm. 4.1:

**Proof of Theorem 4.1**. We have to prove the equality in Eq. (B.50) and show that the terms on the right hand side are as in Thm. 4.1. Using Lem. B.2, more specifically Eq. (B.51), and plugging it in the left hand side of Eq. (B.50) yields

$$\begin{aligned} &\int q(x_0) \left[ \int q(F_M, x_t, \dots, x_1|x_0) dF_M \right] dx_{0:t-1} \\ &= \int \left( \int q(F_M|x_t, \dots, x_0) dF_M \right) q(x_0) \prod_{t'=1}^{t} q(x_{t'}|x_{t'-1}, \dots, x_0) dx_{0:t-1}, \end{aligned} \tag{B.55}$$

where we pulled all terms not depending on $F_M$ out of the inner integral. As $q(F_M|x_t, \dots, x_0)$ is a properly normalised probability density, the inner integral equals one. According to Lem. B.2 the terms $q(x_{t'}|x_{t'-1}, \dots, x_0)$ have the correct form [Eqs. (4.3) – (4.5)] which already completes the proof. □

What remains to be done is to prove Lem. B.2. The proof uses a similar idea as the proof of Lem. B.1 in Appx. B.1 and is also performed via induction.

**Proof of Lemma B.2**. We begin with the base case of the induction and perform the inductive step afterwards.

**Base case** We need to show that Eq. (B.51) holds for $t = 1$, i.e., that

$$q(F_M, x_1|x_0) = q(F_M|x_1, x_0) q(x_1|x_0) \tag{B.56}$$

with the terms on the RHS given by Eqs. (B.52) – (B.54) and Eqs. (4.3) – (4.5), respectively. In order to do so, we will perform the following steps:

i) In the first step we will show that Eqs. (B.3)-(B.6) are applicable, which will enable us to write Eq. (B.56) as

$$q(F_M, x_1|x_0) = \bar{q}(F_M|x_1, x_0) \bar{q}(x_1|x_0),$$

where we denote with $\bar{q}$ that these distributions have the correct conditional dependencies and are Gaussian but we have not yet checked whether their means and covariances coincide with Eqs. (B.52) – (B.54) or Eqs. (4.3) – (4.5). This will be done in the following steps.

ii) Next, we will show that

$$\bar{q}(x_1|x_0) = q(x_1|x_0).$$

iii) In the final step, we will show that

$$\bar{q}(F_M|x_1, x_0) = q(F_M|x_1, x_0).$$

For step i), we start with the definition of $q(F_M, x_1|x_0)$ in Eq. (B.46):

$$\begin{aligned} q(F_M, x_1|x_0) &= q(F_M)p(x_1|x_0, F_M) \\ &= \mathcal{N}(F_M|m_M, S_M)\,\mathcal{N}\left(x_1 \middle| K_{0M}K_{MM}^{-1}F_M, \sigma_x^2 + k_{00} - K_{0M}K_{MM}^{-1}K_{0M}^\top\right), \end{aligned} \tag{B.57}$$

where we used Eqs. (B.48) and (B.49) in the second step. Next, we note that the requirements in Eq. (B.3) are given for the terms above, where we identify $F_M$ as $y$ and $x_1$ as $x$. Applying Eqs. (B.4)-(B.6) to Eq. (B.57), results in

$$q(F_M, x_1|x_0) = \bar{q}(F_M|x_1, x_0)\bar{q}(x_1|x_0) \tag{B.58}$$

with yet to be determined means and covariances, which concludes the first step.

In step ii), we start with the second term on the RHS in Eq. (B.58) and use Eqs. (B.4) and (B.57), yielding

$$\begin{aligned} &\bar{q}(x_1|x_0) \\ &= \mathcal{N}\left(x_1 \middle| K_{0M}K_{MM}^{-1}m_M, \sigma_x^2 + k_{00} - K_{0m}K_{MM}^{-1}K_{0M}^\top + K_{0m}K_{MM}^{-1}S_MK_{MM}^{-1}K_{0M}^\top\right) && \text{(B.59)} \\ &= \mathcal{N}\left(x_1 \middle| \widetilde{\mu}_0, \widetilde{S}_{0,0}\right) = \mathcal{N}\left(x_1 \middle| \hat{\mu}_1, \hat{\Sigma}_1\right), && \text{(B.60)} \end{aligned}$$

where we used the definitions in Eqs. (4.4)-(4.7) in the last line. Together with the definition in Eq. (4.3), this implies that in fact $\bar{q}(x_1|x_0) = q(x_1|x_0)$.

Finally for step iii), using Eqs. (B.5) and (B.57) for the first term on the RHS in Eq. (B.58) results in

$$\begin{aligned} &\bar{q}(F_M|x_1, x_0) \\ &= \mathcal{N}\left(F_M \middle| m_M + S_MK_{MM}^{-1}K_{0M}^\top\widetilde{S}_{0,0}^{-1}(x_1 - \widetilde{\mu}_0), S_M - S_MK_{MM}^{-1}K_{0M}^\top\widetilde{S}_{0,0}^{-1}K_{0_M}K_{MM}^{-1}S_M\right) \\ &= \mathcal{N}\left(F_M \middle| \hat{\mu}_M^1, \hat{\Sigma}_M^1\right). \end{aligned}$$

In the first line we used that $a + Fb = \widetilde{\mu}_0$ and that $A + FBF^\top = \widetilde{S}_{0,0}$ (by comparing Eqs. (B.59) and (B.60) with Eq. (B.4)). Additionally we used the definitions in Eqs. (B.53) and (B.54) in the last line. Together with the definition in Eq. (B.52), this implies that in fact $\bar{q}(F_M|x_1, x_0) = q(F_M|x_1, x_0)$, concluding step iii) and therefore also the base case of the induction.

**Inductive step** We assume that Lem. B.2 holds for some $t = 1, \ldots, T-1$ (induction assumption) and then need to show that it also holds for $t+1$. That is, assuming that

$$q(F_M, x_t, \ldots, x_1|x_0) = q(F_M|x_t, \ldots, x_0) \prod_{t'=1}^{t} q(x_{t'}|x_{t'-1}, \ldots, x_0), \tag{B.61}$$

holds for some $t$ with the terms on the RHS given by Eqs. (B.52) – (B.54), and Eqs. (4.3) – (4.5), respectively, we need to show that this implies that

$$q(F_M, x_{t+1}, \ldots, x_1|x_0) = q(F_M|x_{t+1}, \ldots, x_0) \prod_{t'=1}^{t+1} q(x_{t'}|x_{t'-1}, \ldots, x_0), \tag{B.62}$$

where the terms are again given by Eqs. (B.52) – (B.54) (but with $t \to t+1$), and Eqs. (4.3) – (4.5), respectively.

The way to show this is very similar to the way we showed the base case, the resulting formulas will only look more complicated and we will need one additional step in the beginning:

o) Starting with the LHS of Eq. (B.62) and its definition in Eq. (B.46), we can regroup the terms as follows:

$$\begin{aligned} q(F_M, x_{t+1}, \ldots, x_1|x_0) &= q(F_M) \prod_{t'=1}^{t+1} p(x_{t'}|x_{t'-1}, F_M) \\ &= p(x_{t+1}|x_t, F_M) \left( q(F_M) \prod_{t'=1}^{t} p(x_{t'}|x_{t'-1}, F_M) \right) \\ &= p(x_{t+1}|x_t, F_M) q(F_M, x_t, \ldots, x_1|x_0), \end{aligned} \tag{B.63}$$

where we identified the terms from Eq. (B.46) in the last step. We can therefore immediately apply the induction assumption [Eq. (B.61)] to the second term in Eq. (B.63), resulting in

$$q(F_M, x_{t+1}, \ldots, x_1|x_0) = p(x_{t+1}|x_t, F_M) q(F_M|x_t, \ldots, x_0) \prod_{t'=1}^{t} q(x_{t'}|x_{t'-1}, \ldots, x_0), \tag{B.64}$$

where the $q(x_t|x_{t-1}, \ldots, x_0)$ terms are given by Eqs. (4.3) – (4.5). Comparing this to what we want to show [Eq. (B.62)], we see that it remains to be shown that

$$p(x_{t+1}|x_t, F_M) q(F_M|x_t, \ldots, x_0) = q(F_M|x_{t+1}, \ldots, x_0) q(x_{t+1}|x_t, \ldots, x_0), \tag{B.65}$$

such that the terms on the RHS are given by Eqs. (B.52) – (B.54), and Eqs. (4.3) – (4.5), respectively. From this point on, we will have to do the exact same steps as in the base case, which we will repeat below with updated indices.

i) In the first step we will show that Eqs. (B.3)-(B.6) are applicable, which will enable us to write the LHS of Eq. (B.65) as

$$p(x_{t+1}|x_t, F_M) q(F_M|x_t, \ldots, x_0) = \bar{q}(F_M|x_{t+1}, \ldots, x_0) \bar{q}(x_{t+1}|x_t, \ldots, x_0). \tag{B.66}$$

ii) Next, we will show that

$$\bar{q}(x_{t+1}|x_t, \ldots, x_0) = q(x_{t+1}|x_t, \ldots, x_0).$$

iii) In the final step, we will show that

$$\bar{q}(F_M|x_{t+1},\dots,x_0) = q(F_M|x_{t+1},\dots,x_0).$$

Let us start with step i): We can use the definition in Eq. (B.49) and Eqs. (B.52)-(B.54) (as part of the induction assumption) to write the terms on the LHS of Eq. (B.66) as

$$p(x_{t+1}|x_t,F_M) = \mathcal{N}\left(x_{t+1}\middle|K_{tM}K_{MM}^{-1}F_M, \sigma_x^2 + k_{tt} - K_{tM}K_{MM}^{-1}K_{tM}^\top\right) \tag{B.67}$$

$$\begin{aligned} q(F_M|x_t,\dots,x_0) = \mathcal{N}\Big(F_M\Big| & m_M + S_M K_{MM}^{-1}\left(K_{0:t-1,M}\right)^\top \widetilde{S}_{0:t-1,0:t-1}^{-1}\left(x_{1:t}-\widetilde{\mu}_{0:t-1}\right), \\ & S_M - S_M K_{MM}^{-1}\left(K_{0:t-1,M}\right)^\top \widetilde{S}_{0:t-1,0:t-1}^{-1} K_{0:t-1,M}K_{MM}^{-1}S_M\Big) \end{aligned} \tag{B.68}$$

Next, we note that the requirements in Eq. (B.3) are given for the terms above, where we identify $F_M$ as $y$ and $x_{t+1}$ as $x$. Applying Eqs. (B.4) – (B.6) to Eqs. (B.67) and (B.68), allows us to write the LHS of Eq. (B.66) as

$$p(x_{t+1}|x_t,F_M)q(F_M|x_t,\dots,x_0) = \bar{q}(F_M|x_{t+1},\dots,x_0)\bar{q}(x_{t+1}|x_t,\dots,x_0) \tag{B.69}$$

with yet to be determined means and covariances. This concludes the first step.

For step ii), we examine the second term on the RHS of Eq. (B.69), which can be obtained using Eqs. (B.3) and (B.4) applied to Eqs. (B.67) and (B.68):

$$\bar{q}(x_{t+1}|x_t,\dots,x_0) = \mathcal{N}\left(x_{t+1}\middle|\bar{m}_{t+1},\bar{\Sigma}_{t+1}\right). \tag{B.70}$$

The mean is given by

$$\begin{aligned} \bar{m}_{t+1} &= K_{tM}K_{MM}^{-1}\left[m_M + S_M K_{MM}^{-1}\left(K_{0:t-1,M}\right)^\top \widetilde{S}_{0:t-1,0:t-1}^{-1}\left(x_{1:t}-\widetilde{\mu}_{0:t-1}\right)\right] \\ &= \widetilde{\mu}_t + \widetilde{S}_{t,0:t-1}\widetilde{S}_{0:t-1,0:t-1}^{-1}\left(x_{1:t}-\widetilde{\mu}_{0:t-1}\right) = \hat{\mu}_{t+1}, \end{aligned} \tag{B.71}$$

where we used the definitions in Eqs. (4.4), (4.6), and (4.7) in the second line. The covariance is given by

$$\begin{aligned} \bar{\Sigma}_{t+1} &= \sigma_x^2 + k_{tt} - K_{tM}K_{MM}^{-1}K_{tM}^\top + K_{tM}K_{MM}^{-1}\times \\ &\quad \left[S_M - S_M K_{MM}^{-1}\left(K_{0:t-1,M}\right)^\top \widetilde{S}_{0:t-1,0:t-1}^{-1}K_{0:t-1,M}K_{MM}^{-1}S_M\right]K_{MM}^{-1}K_{tM}^\top \\ &= \widetilde{S}_{t,t} - \widetilde{S}_{t,0:t-1}\widetilde{S}_{0:t-1,0:t-1}^{-1}\widetilde{S}_{0:t-1,t} = \hat{\Sigma}_{t+1}, \end{aligned} \tag{B.72}$$

where we used the definitions in Eqs. (4.5) – (4.7) in the last line. Taken together, Eqs. (B.70) – (B.72) state that $\bar{q}(x_{t+1}|x_t,\dots,x_0)$ is a Gaussian with mean $\hat{\mu}_{t+1}$ and covariance $\hat{\Sigma}_{t+1}$, i.e., that $\bar{q}(x_{t+1}|x_t,\dots,x_0) = q(x_{t+1}|x_t,\dots,x_0)$ [cf. Eq. (4.3)]. This concludes the second step.

For the last step, step iii), we consider the first term on the RHS of Eq. (B.69), which can be obtained using Eqs. (B.3) and (B.5) applied to Eqs. (B.67) and (B.68):

$$\bar{q}(F_M|x_{t+1},\dots,x_0) = \mathcal{N}\left(F_M\middle|\bar{\mu}_M^{t+1},\bar{\Sigma}_M^{t+1}\right). \tag{B.73}$$

As in the previous step, it remains to be shown that the mean and covariance coincide with $\hat{\mu}_M^{t+1}$ and $\hat{\Sigma}_M^{t+1}$ given in Eq. (B.53) and (B.54), respectively. Here we can build on the previous experience with the similar proof in Appx. B.1, since we require exactly the same steps (although with different

quantities) as in Eqs. (B.40) – (B.44). Therefore we only sketch the derivation exemplarily for $\bar{\mu}_M^{t+1}$ here: Starting from

$$\begin{aligned}\bar{\mu}_M^{t+1} = m_M + S_M K_{MM}^{-1} \left(K_{0:t-1,M}\right)^\top \widetilde{S}_{0:t-1,0:t-1}^{-1} \left(x_{1:t} - \widetilde{\mu}_{0:t-1}\right) + \\ \left[S_M - S_M K_{MM}^{-1} \left(K_{0:t-1,M}\right)^\top \widetilde{S}_{0:t-1,0:t-1}^{-1} K_{0:t-1,M} K_{MM}^{-1} S_M\right] \times \\ K_{MM}^{-1} K_{tM}^\top \hat{\Sigma}_{t+1}^{-1} \left[x_{t+1} - \widetilde{\mu}_t - \widetilde{S}_{t,0:t-1} \widetilde{S}_{0:t-1,0:t-1}^{-1} \left(x_{1:t} - \widetilde{\mu}_{0:t-1}\right)\right],\end{aligned} \tag{B.74}$$

where $\hat{\Sigma}_{t+1}$ is as in Eq. (B.72), we can reorder the terms until we get to a point where we can apply the block matrix inversion formula [Eq. (B.1)] backwards. In order to do so, we identify the terms of Eq. (B.1) as

$$A = \widetilde{S}_{0:t-1,0:t-1}, \qquad B = \widetilde{S}_{0:t-1,t}, \qquad C = \widetilde{S}_{t,0:t-1}, \qquad \widetilde{D} = \hat{\Sigma}_{t+1}^{-1},$$

and therefore $D = \widetilde{S}_{t,t}$ [by comparing the definition of $\widetilde{D}$ in Eq. (B.2) with $\hat{\Sigma}_{t+1}^{-1}$ in Eq. (B.72)]. This results (after several steps) in

$$\bar{\mu}_M^{t+1} = m_M + S_M K_{MM}^{-1} \begin{pmatrix}\left(K_{0:t-1,M}\right)^\top & K_{tM}^\top\end{pmatrix} \begin{pmatrix}\widetilde{S}_{0:t-1,0:t-1} & \widetilde{S}_{0:t-1,t} \\ \widetilde{S}_{t,0:t-1} & \widetilde{S}_{t,t}\end{pmatrix}^{-1} \begin{pmatrix}x_{1:t} - \widetilde{\mu}_{0:t-1} \\ x_{t+1} - \widetilde{\mu}_t\end{pmatrix} \tag{B.75}$$

$$= m_M + S_M K_{MM}^{-1} \left(K_{0:t,M}\right)^\top \widetilde{S}_{0:t,0:t}^{-1} \left(x_{1:t+1} - \widetilde{\mu}_{0:t}\right) = \hat{\mu}_M^{t+1}, \tag{B.76}$$

where we first rewrote the first line according to our slicing notation and then applied the definition in Eq. (B.53) in the last step. Doing the same for the covariance, we can similarly show that $\bar{\Sigma}_M^{t+1} = \hat{\Sigma}_M^{t+1}$. Hence $\bar{q}(F_M|x_{t+1}, \ldots, x_0) = q(F_M|x_{t+1}, \ldots, x_0)$. This concludes step iii) as well as the induction step and therefore the proof of Lem. B.2. □

## B.3 The FITC approximation for GPSSMs

The aim of this section is to prove Prop. 4.2. We will do this by pointing out the differences in the derivation of the original ELBO of the VCDT methods which is done in Appx. A.2.4. A direct derivation can be found in Lindinger et al. (2022, Appx. A).

**Proof of Proposition 4.2**. Plugging in the FITC approximation [Eq. (4.11)] in the prior and the approximate posterior in Eqs. (2.26) and (2.34) yields

$$p_{\text{FITC}}(Y_T, X_{T_0}, F_T, F_M) = p(x_0)p(F_M) \prod_{t=1}^{T} p(y_t|x_t)p(x_t|f_{t-1}, x_{t-1})p(f_{t-1}|x_{t-1}, F_M), \tag{B.77}$$

$$q_{\text{FITC}}(X_{T_0}, F_T, F_M) = q(x_0)q(F_M) \prod_{t=0}^{T-1} p(f_t|x_t, F_M)q(x_{t+1}|F_M, x_t), \tag{B.78}$$

where the definitions of the individual terms can be found throughout the thesis (especially Sec. 2.4 and Appx. A.2.4). However they are not important for the proof at hand, only the conditional dependencies matter in this case.

In the following we perform the same steps as in Appx. A.2.4, only replacing $p(Y_T, X_{T_0}, F_T, F_M)$ by Eq. (B.77) and $q(X_{T_0}, F_T, F_M)$ by Eq. (B.78). Plugging Eqs. (B.77) and (B.78) in Eq. (A.12) yields essentially the same result as Eq. (A.27) since the FITC terms within the expectation cancel.

The only difference is that the expectation is over $q_{\text{FITC}}$ instead of the standard variational posterior. The FITC posterior has no influence in the next step [Eq. (A.28)], where the KL-divergences for $x_0$ and the $F_M$ emerge. Continuing with the expectation over the data fit term in Eq. (A.29), we see that the FITC posterior in Eq. (B.78) makes no difference to the result since the data fit term is independent of any $f_t$ such that the exact form of $p(F_T|X_T, F_M)$ does not play a role here. Only the last term [Eq. (A.31)] requires a more careful consideration: Replacing $p(F_T|X_T, F_M)$ by $\prod_{t=0}^{T-1} p(f_t|x_t, F_M)$ in the first step, i.e., plugging in the FITC approximation, makes it even easier to see that the integration over $df_{0:T-1}$ will leave only the $p(f_{t-1}|x_{t-1}, F_M)$ term due to the property in Eq. (A.9) (see also Lindinger et al., 2022, Eq. (33)). The rest of Eq. (A.31) remains the same.

We have therefore shown that all terms in the ELBO for the VCDT method in Eq. (A.27) stay the same when the FITC approximation is used. This completes the proof. □

## B.4 Equivalence of SDE GPSSMs and canonical GPSSMs

The goal of this section is to provide a complete proof for Thm. 5.1. As discussed in the sketch of the proof in Sec. 5.2.2, the only non-trivial step in this proof consists of showing the equivalence of $q^{\Delta}(x_j)$ and $q(x_t)$ in Eqs. (2.31) and (5.21), respectively for $R = 1$ and $j = t$. For $q(x_t)$ we already have a formula in Thm. 4.1 which changes very little if the residual transition model in Eq. (5.18) is used:

**Remark B.3.** *When using the residual transition model in Eq.* (5.18) *instead of the usual transition model in Eq.* (2.28)*, the only change in Thm. 4.1 is that Eq.* (4.6) *is replaced by*

$$\widetilde{\mu}_t = x_t + K_{tM} K_{MM}^{-1} m_M. \tag{B.79}$$

This can be easily seen by checking the parts in the proof of Thm. 4.1 in Appx. B.2 where the mean of the transition model plays a role (see also Longi et al., 2021, Appx. A for the complete proof with the residual transition model).

For $q^{\Delta}(x_j)$ we only have the general formula in Eq. (5.21) but we can naturally perform an equivalent marginalisation of the inducing outputs $F_M$ as for $q(x_t)$ which we summarise in the following lemma (Longi et al., 2021):

**Lemma B.4.** *For the variational posterior of the SDE formulation of the GPSSM given in Eq.* (5.19) *the marginals of the latent state at time point $j \in \{1, \dots, J\}$ in Eq.* (5.21)*, can be obtained as*

$$q^{\Delta}(x_j) = \int q^{\Delta}(x_0) \left[ \prod_{j'=1}^{j} q^{\Delta}(x_{j'}|x_{j'-1}, \dots, x_0) \right] dx_{0:j-1}, \tag{B.80}$$

*where all terms are Gaussian:*

$$q^{\Delta}(x_j|x_{j-1}, \dots, x_0) = \mathcal{N}\left(x_j \middle| \hat{\mu}_j, \hat{\Sigma}_j\right), \tag{B.81}$$

$$\hat{\mu}_j = \widetilde{\mu}_{j-1} + \widetilde{S}_{j-1,0:j-2} \widetilde{S}_{0:j-2,0:j-2}^{-1} \left(x_{1:j-1} - \widetilde{\mu}_{0:j-2}\right), \tag{B.82}$$

$$\hat{\Sigma}_j = \widetilde{S}_{j-1,j-1} - \widetilde{S}_{j-1,0:j-2} \widetilde{S}_{0:j-2,0:j-2}^{-1} \widetilde{S}_{0:j-2,j-1}. \tag{B.83}$$

*Here, the terms are given by*

$$\widetilde{\mu}_j = x_j + R\Delta_t K_{jM}^{\Delta} \left(K_{MM}^{\Delta}\right)^{-1} m_M^{\Delta}, \tag{B.84}$$

$$\begin{aligned}\widetilde{S}_{j,j'} = &(R\Delta_t)^2 K_{jM}^{\Delta} \left(K_{MM}^{\Delta}\right)^{-1} S_M^{\Delta} \left(K_{MM}^{\Delta}\right)^{-1} {K_{j'M}^{\Delta}}^{\top} \\ &+ \delta_{jj'} \left[R\Delta_t \sigma_{\Delta}^2 + (R\Delta_t)^2 \left(k_{jj}^{\Delta} - K_{jM}^{\Delta} \left(K_{MM}^{\Delta}\right)^{-1} {K_{j'M}^{\Delta}}^{\top}\right)\right].\end{aligned} \tag{B.85}$$

With this lemma, which we will prove below, we can finally provide a complete proof of Thm. 5.1:

**Proof of Theorem 5.1**. The idea of the proof is to show the equivalence in Eq. (5.22) by demonstrating that all terms in the ELBOs are equal with the settings provided in Thm. 5.1. First, it is easy to see that the KL-terms in Eqs. (2.30) and (5.20) are equal since the respective distributions in both ELBOs are assumed to be the same. It remains to be shown that

$$\sum_{j=1}^{J} \mathbb{E}_{q^{\Delta}(x_j)} \left[\log p^{\Delta}(y_j|x_j)\right] = \sum_{t=1}^{T} \mathbb{E}_{q(x_t)} \left[\log p(y_t|x_t)\right].$$

By setting $R = 1$ (as required by Thm. 5.1) we have $J = T/R = T$ and therefore equally many terms in both sums. Additionally the observation model $p^{\Delta}(y_j|x_j)$ is assumed to be the same as $p(y_t|x_t)$. Hence it only remains to be shown that $q^{\Delta}(x_j) = q(x_t)$ for $j = t$. Comparing the formulas for $q(x_t)$ provided in Thm. 4.1 and Rem. B.3 with those for $q^{\Delta}(x_j)$ in Lem. B.4, we see that the settings provided in Thm. 5.1 [especially Eqs. (5.23)–(5.25) but also $m_M^{\Delta} = m_M$ and $S_M^{\Delta} = S_M$] lead to the marginals being equal for $j = t$. This completes the proof. □

What remains is to prove Lem. B.4. Due to its similarity to Thm. 4.1, the proof naturally works in the same way, i.e., with the help of an auxiliary lemma that has to be proved via induction. The latter is the equivalent of Lem. B.2 and has the following form for the SDE GPSSM:

**Lemma B.5.** *The integrand of the inner integral in Eq.* (5.21) *can be equally written as*

$$q^{\Delta}(F_M) \prod_{j'=0}^{j-1} p^{\Delta}(x_{j'+1}|x_{j'}, F_M) = q^{\Delta}(F_M|x_j, \ldots, x_0) \prod_{j'=1}^{j} q^{\Delta}(x_{j'}|x_{j'-1}, \ldots, x_0), \tag{B.86}$$

*for* $j \in \{1, \ldots, J\}$. *Here, the* $q^{\Delta}(x_j|x_{j-1}, \ldots, x_0)$ *are as in Eqs.* (B.81)-(B.83) *and*

$$q^{\Delta}(F_M|x_j, \ldots, x_0) = \mathcal{N}\left(F_M \middle| \hat{\mu}_M^j, \hat{\Sigma}_M^j\right) \tag{B.87}$$

*with [see Eqs.* (B.84) *and* (B.85) *for the definitions of* $\widetilde{\mu}$ *and* $\widetilde{S}$*]*

$$\hat{\mu}_M^j = m_M^{\Delta} + (R\Delta_t) S_M^{\Delta} \left(K_{MM}^{\Delta}\right)^{-1} \left(K_{0:j-1,M}^{\Delta}\right)^{\top} \widetilde{S}_{0:j-1,0:j-1}^{-1} \left(x_{1:j} - \widetilde{\mu}_{0:j-1}\right), \tag{B.88}$$

$$\hat{\Sigma}_M^j = S_M^{\Delta} - (R\Delta_t)^2 S_M^{\Delta} \left(K_{MM}^{\Delta}\right)^{-1} \left(K_{0:j-1,M}^{\Delta}\right)^{\top} \widetilde{S}_{0:j-1,0:j-1}^{-1} K_{0:j-1,M}^{\Delta} \left(K_{MM}^{\Delta}\right)^{-1} S_M^{\Delta}. \tag{B.89}$$

With Lem. B.5, the proof of Lem. B.4 is very simple:

**Proof of Lemma B.4**. The proof can be performed in exactly the same way as the proof for Thm. 4.1 in Appx. B.2: We have to prove the equality in Eq. (B.80) and show that the terms on the right hand

side are as in Lem. B.4. Using Lem. B.5, more specifically Eq. $(\mathrm{B}.86)$, and plugging it in the general formula for $q^{\Delta}(x_j)$ in Eq. (5.21) yields

$$\begin{aligned} q^{\Delta}(x_j) &= \int q^{\Delta}(x_0) \left[ \int q^{\Delta}(F_M) \prod_{j'=0}^{j-1} p^{\Delta}(x_{j'+1}|x_{j'}, F_M) dF_M \right] dx_{0:t-1} \\ &= \int \left( \int q^{\Delta}(F_M|x_j, \ldots, x_0) dF_M \right) q^{\Delta}(x_0) \prod_{j'=1}^{j} q^{\Delta}(x_{j'}|x_{j'-1}, \ldots, x_0) dx_{0:j-1}, \end{aligned}$$

where we pulled all terms not depending on $F_M$ out of the inner integral. As $q^{\Delta}(F_M|x_j, \ldots, x_0)$ is a properly normalised probability density, the inner integral equals one. According to Lem. B.5 the terms $q^{\Delta}(x_{j'}|x_{j'-1}, \ldots, x_0)$ have the correct form [Eqs. (B.81) – (B.83)] which already completes the proof. □

Finally, there remains the proof of Lem. B.5. This is the SDE GPSSM equivalent of Lem. B.2 which concerned canonical GPSSMs. The proof naturally works in the same way:

**Proof of Lemma B.5**. The proof can be performed in exactly the same way as the proof for Lem. B.2 in Appx. B.2. Care has to be taken that the factors of $R\Delta_t$ appearing in Eqs. (B.84), (B.85) and (5.15) are treated correctly. The only difference apart from this is that the definitions of all quantities from Appx. B.2 have to be replaced with their counterparts in Sec. 5.2.2. □

# Bibliography


Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al. (2016). Tensorflow: A system for large-scale machine learning. *USENIX Symposium on Operating Systems Design and Implementation*.

Abdar, M., Pourpanah, F., Hussain, S., Rezazadegan, D., Liu, L., Ghavamzadeh, M., Fieguth, P., Cao, X., Khosravi, A., Acharya, U. R., Makarenkov, V., & Nahavandi, S. (2021). A review of uncertainty quantification in deep learning: Techniques, applications and challenges. *Inf. Fusion*, **76**, 243–297.

Adam, V., Eleftheriadis, S., Artemev, A., Durrande, N., & Hensman, J. (2020). Doubly sparse variational Gaussian processes. *International Conference on Artificial Intelligence and Statistics*.

Adam, V., Chang, P., & Khan, M. E. (2021). Dual parameterization of sparse variational Gaussian processes. *Advances in Neural Information Processing Systems*.

Adams, R. P., & Stegle, O. (2008). Gaussian process product models for nonparametric nonstationarity. *International Conference on Machine Learning*.

Adlam, B., Snoek, J., & Smith, S. L. (2020). Cold posteriors and aleatoric uncertainty. *International Conference on Machine Learning: Workshop on Uncertainty and Robustness in Deep Learning*.

Agrawal, D., Papamarkou, T., & Hinkle, J. D. (2020). Wide neural networks with bottlenecks are deep Gaussian processes. *J. Mach. Learn. Res.*, **21**, 175:1–175:66.

Aicher, C., Ma, Y.-A., Foti, N. J., & Fox, E. B. (2019). Stochastic gradient MCMC for state space models. *SIAM J. Math. Data Sci.*, **1**(3), 555–587.

Aitchison, L. (2021). A statistical theory of cold posteriors in deep neural networks. *International Conference on Learning Representations*.

Álvarez, M. A., & Lawrence, N. D. (2011). Computationally efficient convolved multiple output Gaussian processes. *J. Mach. Learn. Res.*, **12**, 1459–1500.

Álvarez, M. A., Rosasco, L., & Lawrence, N. D. (2012). Kernels for vector-valued functions: A review. *Found. Trends Mach. Learn.*, **4**(3), 195–266.

Amari, S.-I. (1998). Natural gradient works efficiently in learning. *Neural Comput.*, **10**, 251–276.

Amodei, D., Olah, C., Steinhardt, J., Christiano, P. F., Schulman, J., & Mané, D. (2016). Concrete problems in AI safety. *arXiv preprint arXiv:1606.06565*.

Bauer, M., van der Wilk, M., & Rasmussen, C. E. (2016). Understanding probabilistic sparse Gaussian process approximations. *Advances in Neural Information Processing Systems*.

Bell, B. M. (2000). The marginal likelihood for parameters in a discrete Gauss-Markov process. *IEEE Trans. Signal Process.*, **48**(3), 870–873.

Bishop, C. M. (2006). *Pattern recognition and machine learning*. Springer.

Bitzer, M., Meister, M., & Zimmer, C. (2022). Structural kernel search via Bayesian optimization and symbolical optimal transport. *arXiv preprint arXiv:2210.11836*.

Blei, D., Kucukelbir, A., & McAuliffe, J. (2017). Variational inference: A review for statisticians. *J. Amer. Stat. Assoc.*

Blomqvist, K., Kaski, S., & Heinonen, M. (2019). Deep convolutional Gaussian processes. *European Conference on Machine Learning and Knowledge Discovery in Databases*.

Bonilla, E. V., Chai, K. M. A., & Williams, C. K. I. (2007). Multi-task Gaussian process prediction. *Advances in Neural Information Processing Systems*.

Bottou, L. (2010). Large-scale machine learning with stochastic gradient descent. *International Conference on Computational Statistics*.

Breiman, L., & Friedman, J. H. (1985). Estimating optimal transformations for multiple regression and correlation. *J. Amer. Stat. Assoc.*, **80**(391), 580–598.

Briol, F.-X., Oates, C. J., Girolami, M., Osborne, M. A., & Sejdinovic, D. (2019). Probabilistic integration: A role in statistical computation? *Stat. Sci.*, **34**(1), 1–22.

Brüdigam, J., Schuck, M., Capone, A., Sosnowski, S., & Hirche, S. (2022). Structure-preserving learning using Gaussian processes and variational integrators. *Annual Learning for Dynamics and Control Conference*.

Bui, T., Hernández-Lobato, D., Hernandez-Lobato, J., Li, Y., & Turner, R. (2016). Deep Gaussian processes for regression using approximate expectation propagation. *International Conference on Machine Learning*.

Bui, T. D., Nguyen, C. V., & Turner, R. E. (2017). Streaming sparse Gaussian process approximations. *Advances in Neural Information Processing Systems*.

Burt, D. R., Rasmussen, C. E., & van der Wilk, M. (2019). Rates of convergence for sparse variational Gaussian process regression. *International Conference on Machine Learning*.

Calandra, R., Peters, J., Rasmussen, C. E., & Deisenroth, M. P. (2016). Manifold Gaussian processes for regression. *International Joint Conference on Neural Networks*.

Chen, T., Fox, E., & Guestrin, C. (2014). Stochastic gradient Hamiltonian Monte Carlo. *International Conference on Machine Learning*.

Chen, R. T. Q., Rubanova, Y., Bettencourt, J., & Duvenaud, D. (2018). Neural ordinary differential equations. *Advances in Neural Information Processing Systems*.

Cheng, C.-A., & Boots, B. (2016). Incremental variational sparse Gaussian process regression. *Advances in Neural Information Processing Systems*.

Chung, J., Gülçehre, Ç., Cho, K., & Bengio, Y. (2014). Empirical evaluation of gated recurrent neural networks on sequence modeling. *Advances in Neural Information Processing Systems: Workshop on Deep Learning and Representation Learning*.

Csató, L., & Opper, M. (2002). Sparse on-line Gaussian processes. *Neural Comput.*, **14**(3), 641–668.

Curi, S., Melchior, S., Berkenkamp, F., & Krause, A. (2020). Structured variational inference in partially observable unstable Gaussian process state space models. *Annual Learning for Dynamics and Control Conference*.

Cutajar, K., Bonilla, E. V., Michiardi, P., & Filippone, M. (2017). Random feature expansions for deep Gaussian processes. *International Conference on Machine Learning*.

Dai, Z., Damianou, A., González, J., & Lawrence, N. D. (2016). Variational auto-encoded deep Gaussian processes. *International Conference on Learning Representations*.

Daley, R. (1993). *Atmospheric data analysis*. Cambridge University Press.

Damianou, A. C., Titsias, M. K., & Lawrence, N. D. (2011). Variational Gaussian process dynamical systems. *Advances in Neural Information Processing Systems*.

Damianou, A. C., & Lawrence, N. D. (2013). Deep Gaussian processes. *International Conference on Artificial Intelligence and Statistics*.

Dandekar, R., Chung, K., Dixit, V., Tarek, M., Garcia-Valadez, A., Vemula, K. V., & Rackauckas, C. (2020). Bayesian neural ordinary differential equations. *arXiv preprint arXiv:2012.07244*.

Datta, A., Banerjee, S., Finley, A. O., & Gelfand, A. E. (2016). On nearest-neighbor Gaussian process models for massive spatial data. *WIREs Comput. Stat.*, **8**(5), 162–171.

Deisenroth, M. P., & Rasmussen, C. E. (2011). PILCO: A model-based and data-efficient approach to policy search. *International Conference on Machine Learning*.

Deisenroth, M. P., & Ng, J. W. (2015). Distributed Gaussian processes. *International Conference on Machine Learning*.

Doerr, A., Daniel, C., Schiegg, M., Nguyen-Tuong, D., Schaal, S., Toussaint, M., & Trimpe, S. (2018). Probabilistic recurrent state-space models. *International Conference on Machine Learning*.

Duncker, L., Bohner, G., Boussard, J., & Sahani, M. (2019). Learning interpretable continuous-time models of latent stochastic dynamical systems. *International Conference on Machine Learning*.

Durrande, N., Adam, V., Bordeaux, L., Eleftheriadis, S., & Hensman, J. (2019). Banded matrix operators for Gaussian Markov models in the automatic differentiation era. *International Conference on Artificial Intelligence and Statistics*.

Dutordoir, V., van der Wilk, M., Artemev, A., & Hensman, J. (2020). Bayesian image classification with deep convolutional Gaussian processes. *International Conference on Artificial Intelligence and Statistics*.

Dutordoir, V., Hensman, J., van der Wilk, M., Ek, C. H., Ghahramani, Z., & Durrande, N. (2021). Deep neural networks as point estimates for deep Gaussian processes. *Advances in Neural Information Processing Systems*.

Duvenaud, D., Lloyd, J., Grosse, R., Tenenbaum, J., & Zoubin, G. (2013). Structure discovery in nonparametric regression through compositional kernel search. *International Conference on Machine Learning*, (3).

Duvenaud, D. (2014). *Automatic model construction with Gaussian processes* (Doctoral dissertation). University of Cambridge, UK.

Duvenaud, D., Rippel, O., Adams, R. P., & Ghahramani, Z. (2014). Avoiding pathologies in very deep networks. *International Conference on Artificial Intelligence and Statistics*.

Eleftheriadis, S., Nicholson, T., Deisenroth, M., & Hensman, J. (2017). Identification of Gaussian process state space models. *Advances in Neural Information Processing Systems*.

Ensinger, K., Solowjow, F., Ziesche, S., Tiemann, M., & Trimpe, S. (2021). Structure-preserving Gaussian process dynamics. *arXiv preprint arXiv:2102.01606*.

Eriksson, D., Pearce, M., Gardner, J., Turner, R. D., & Poloczek, M. (2019). Scalable global optimization via local Bayesian optimization. *Advances in Neural Information Processing Systems*.

Fortuin, V., Garriga-Alonso, A., Wenzel, F., Rätsch, G., Turner, R., van der Wilk, M., & Aitchison, L. (2022). Bayesian neural network priors revisited. *International Conference on Learning Representations*.

Frigola, R., Lindsten, F., Schön, T. B., & Rasmussen, C. E. (2013). Bayesian inference and learning in Gaussian process state-space models with particle MCMC. *Advances in Neural Information Processing Systems*.

Frigola, R., Chen, Y., & Rasmussen, C. E. (2014). Variational Gaussian process state-space models. *Advances in Neural Information Processing Systems*.

Frigola, R. (2015). *Bayesian time series learning with Gaussian processes* (Doctoral dissertation). University of Cambridge, UK.

Frommater, S. (2018). *Phenomenological modelling of particulate emissions in direct injection spark ignition engines for driving cycle simulations* (Doctoral dissertation). Technische Universität Darmstadt, Germany.

Girard, A., Rasmussen, C. E., & Murray-Smith, R. (2003). Multiple-step ahead prediction for non linear dynamic systems: A Gaussian process treatment with propagation of the uncertainty. *Advances in Neural Information Processing Systems*.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (2014). Generative adversarial nets. *Advances in Neural Information Processing Systems*.

Goodfellow, I. J., Bengio, Y., & Courville, A. C. (2016). *Deep learning*. MIT Press.

Gould, S., Hartley, R., & Campbell, D. (2019). Deep declarative networks: A new hope. *arXiv preprint arXiv:1909.04866*.

Gunter, T., Osborne, M. A., Garnett, R., Hennig, P., & Roberts, S. J. (2014). Sampling for inference in probabilistic models with fast Bayesian quadrature. *Advances in Neural Information Processing Systems*.

Hamelijnck, O., Damoulas, T., Wang, K., & Girolami, M. A. (2019). Multi-resolution multi-task Gaussian processes. *Advances in Neural Information Processing Systems*.

Haußmann, M., Gerwinn, S., Look, A., Rakitsch, B., & Kandemir, M. (2021). Learning partially known stochastic dynamics with empirical PAC Bayes. *International Conference on Artificial Intelligence and Statistics*.

Havasi, M., Hernández-Lobato, J. M., & Murillo-Fuentes, J. J. (2018). Inference in deep Gaussian processes using stochastic gradient Hamiltonian Monte Carlo. *Advances in Neural Information Processing Systems*.

Hayashi, K., Imaizumi, M., & Yoshida, Y. (2020). On random subsampling of Gaussian process regression: A graphon-based analysis. *International Conference on Artificial Intelligence and Statistics*.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. *Conference on Computer Vision and Pattern Recognition*.

Heaton, J. B., Polson, N. G., & Witte, J. H. (2017). Deep learning for finance: Deep portfolios. *Appl. Stoch. Models Bus. Ind.*, **33**(1), 3–12.

Hebbal, A., Brevault, L., Balesdent, M., Talbi, E.-G., & Melab, N. (2021). Bayesian optimization using deep Gaussian processes with applications to aerospace system design. *Optim. Eng.*, **22**, 321–361.

Hegde, P., Heinonen, M., Lähdesmäki, H., & Kaski, S. (2019). Deep learning with differential Gaussian process flows. *International Conference on Artificial Intelligence and Statistics*.

Hegde, P., Yıldız, Ç., Lähdesmäki, H., Kaski, S., & Heinonen, M. (2022). Variational multiple shooting for Bayesian ODEs with Gaussian processes. *Conference on Uncertainty in Artificial Intelligence*.

Heinonen, M., Yildiz, C., Mannerström, H., Intosalmi, J., & Lähdesmäki, H. (2018). Learning unknown ODE models with Gaussian processes. *International Conference on Machine Learning*.

Hensman, J., Fusi, N., & Lawrence, N. D. (2013). Gaussian processes for big data. *Conference on Uncertainty in Artificial Intelligence*.

Hensman, J., & Lawrence, N. D. (2014). Nested variational compression in deep Gaussian processes. *arXiv preprint arXiv:1412.1370*.

Hensman, J., Matthews, A. G. d. G., Filippone, M., & Ghahramani, Z. (2015). MCMC for variationally sparse Gaussian processes. *Advances in Neural Information Processing Systems*.

Hensman, J., Durrande, N., & Solin, A. (2017). Variational Fourier features for Gaussian processes. *J. Mach. Learn. Res.*, **18**, 151:1–151:52.

Hewing, L., Arcari, E., Fröhlich, L. P., & Zeilinger, M. N. (2020). On simulation and trajectory prediction with Gaussian process dynamics. *Annual Learning for Dynamics and Control Conference*.

Hinton, G. E. (2002). Training products of experts by minimizing contrastive divergence. *Neural Comput.*, **14**(8), 1771–1800.

Hochreiter, S., & Schmidhuber, J. (1996). LSTM can solve hard long time lag problems. *Advances in Neural Information Processing Systems*.

Ialongo, A. D., van der Wilk, M., Hensman, J., & Rasmussen, C. E. (2019). Overcoming mean-field approximations in recurrent Gaussian process models. *International Conference on Machine Learning*.

Izmailov, P., Novikov, A., & Kropotov, D. (2018). Scalable Gaussian processes with billions of inducing inputs via tensor train decomposition. *International Conference on Artificial Intelligence and Statistics*.

Izmailov, P., Vikram, S., Hoffman, M. D., & Wilson, A. G. G. (2021). What are Bayesian neural network posteriors really like? *International Conference on Machine Learning*.

Jørgensen, M., Deisenroth, M. P., & Salimbeni, H. (2020). Stochastic differential equations with variational Wishart diffusions. *International Conference on Machine Learning*.

Jospin, L. V., Laga, H., Boussaid, F., Buntine, W., & Bennamoun, M. (2022). Hands-on Bayesian neural networks—a tutorial for deep learning users. *IEEE Comput. Intell. Mag.*, **17**(2), 29–48.

Journel, A., & Huijbregts, C. (2003). *Mining geostatistics*. Blackburn Press.

Kaiser, M., Otte, C., Runkler, T. A., & Ek, C. H. (2018). Bayesian alignments of warped multi-output Gaussian processes. *Advances in Neural Information Processing Systems*.

Kalman, R. E. (1960). A new approach to linear filtering and prediction problems. *Trans. ASME, J. Basic Eng.*, **82**, 35–45.

Kim, H.-M., Mallick, B. K., & Holmes, C. C. (2005). Analyzing nonstationary spatial data using piecewise Gaussian processes. *J. Amer. Stat. Assoc.*, **100**(470), 653–668.

Kingma, D. P., & Welling, M. (2014). Auto-encoding variational Bayes. *International Conference on Learning Representations*.

Kingma, D. P., & Ba, J. (2015). Adam: A method for stochastic optimization. *International Conference on Learning Representations*.

Kingma, D. P., Salimans, T., & Welling, M. (2015). Variational dropout and the local reparameterization trick. *Advances in Neural Information Processing Systems*.

Kocijan, J., Girard, A., Banko, B., & Murray-Smith, R. (2005). Dynamic systems identification with Gaussian processes. *Math. Comput. Model. Dyn. Syst.*, **11**(4), 411–424.

Koulaei, M. H., & Toutounian, F. (2007). On computing of block ILU preconditioner for block tridiagonal systems. *J. Comput. Appl. Math.*, **202**(2), 248–257.

Krantz, S. G., & Parks, H. R. (2002). *The implicit function theorem: History, theory, and applications*. Springer Science & Business Media.

Krause, A., Singh, A. P., & Guestrin, C. (2008). Near-optimal sensor placements in Gaussian processes: Theory, efficient algorithms and empirical studies. *J. Mach. Learn. Res.*, **9**, 235–284.

Kristensen, K., Nielsen, A., Berg, C. W., Skaug, H., & Bell, B. M. (2016). TMB : Automatic differentiation and Laplace approximation. *J. Stat. Softw.*, **70**(5), 1–21.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems*.

Lakshminarayanan, B., Pritzel, A., & Blundell, C. (2017). Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in Neural Information Processing Systems*.

Lawrence, N. D. (2005). Probabilistic non-linear principal component analysis with Gaussian process latent variable models. *J. Mach. Learn. Res.*, **6**, 1783–1816.

Lázaro-Gredilla, M., & Figueiras-Vidal, A. (2009). Inter-domain Gaussian processes for sparse inference using inducing features. *Advances in Neural Information Processing Systems*.

Li, X., Wong, T.-K. L., Chen, R. T. Q., & Duvenaud, D. (2020). Scalable gradients for stochastic differential equations. *International Conference on Artificial Intelligence and Statistics*.

Lindinger, J., Reeb, D., Lippert, C., & Rakitsch, B. (2020). Beyond the mean-field: Structured deep Gaussian processes improve the predictive uncertainties. *Advances in Neural Information Processing Systems*.

Lindinger, J., Rakitsch, B., & Lippert, C. (2022). Laplace approximated Gaussian process state-space models. *Conference on Uncertainty in Artificial Intelligence*.

Lipton, Z. C., Kale, D. C., Elkan, C., & Wetzel, R. C. (2016). Learning to diagnose with LSTM recurrent neural networks. *International Conference on Learning Representations*.

Liu, H., Ong, Y.-S., Shen, X., & Cai, J. (2020). When Gaussian process meets big data: A review of scalable GPs. *IEEE Trans. Neural Networks Learn. Syst.*, **31**(11), 4405–4423.

Longi, K., Lindinger, J., Duennbier, O., Kandemir, M., Klami, A., & Rakitsch, B. (2021). Traversing time with multi-resolution Gaussian process state-space models. *arXiv preprint arXiv:2112.03230*.

Longi, K., Lindinger, J., Duennbier, O., Kandemir, M., Klami, A., & Rakitsch, B. (2022). Traversing time with multi-resolution Gaussian process state-space models. *Annual Learning for Dynamics and Control Conference*.

Look, A., & Kandemir, M. (2019). Differential Bayesian neural nets. *Advances in Neural Information Processing Systems: Workshop on Bayesian Deep Learning*.

MacKay, D. (2003). *Information theory, inference and learning algorithms*. Cambridge University Press.

Maddox, W. J., Izmailov, P., Garipov, T., Vetrov, D. P., & Wilson, A. G. (2019). A simple baseline for Bayesian uncertainty in deep learning. *Advances in Neural Information Processing Systems*.

Maddox, W. J., Potapczynski, A., & Wilson, A. G. (2022). Low-precision arithmetic for fast Gaussian processes. *Conference on Uncertainty in Artificial Intelligence*.

Margossian, C., Vehtari, A., Simpson, D., & Agrawal, R. (2020). Hamiltonian Monte Carlo using an adjoint-differentiated Laplace approximation: Bayesian inference for latent Gaussian models and beyond. *Advances in Neural Information Processing Systems*.

Martens, J., & Grosse, R. B. (2015). Optimizing neural networks with Kronecker-factored approximate curvature. *International Conference on Machine Learning*.

Martinez, J., Black, M. J., & Romero, J. (2017). On human motion prediction using recurrent neural networks. *Conference on Computer Vision and Pattern Recognition*.

Matheron, G. (1973). The intrinsic random functions and their applications. *Adv. Appl. Probab.*, **5**(3), 439–468.

Matthews, A. G. d. G., Hensman, J., Turner, R. E., & Ghahramani, Z. (2016). On sparse variational methods and the Kullback-Leibler divergence between stochastic processes. *International Conference on Artificial Intelligence and Statistics*.

Mattos, C. L. C., Dai, Z., Damianou, A. C., Forth, J., Barreto, G. A., & Lawrence, N. D. (2016). Recurrent Gaussian processes. *International Conference on Learning Representations*.

McIntire, M., Ratner, D., & Ermon, S. (2016). Sparse Gaussian processes for Bayesian optimization. *Conference on Uncertainty in Artificial Intelligence*.

Mikheeva, O., Kazlauskaite, I., Hartshorne, A., Kjellström, H., Ek, C. H., & Campbell, N. D. F. (2022). Aligned multi-task Gaussian process. *International Conference on Artificial Intelligence and Statistics*.

Murray, I., & Adams, R. P. (2010). Slice sampling covariance hyperparameters of latent Gaussian models. *Advances in Neural Information Processing Systems*.

Neal, R. M. (1993). Probabilistic inference using Markov chain Monte Carlo methods. *Technical Report, University of Toronto*.

Neal, R. M. (1996). *Bayesian learning for neural networks*. Springer New York.

Nguyen, T. V., & Bonilla, E. V. (2014). Collaborative multi-output Gaussian processes. *Conference on Uncertainty in Artificial Intelligence*.

Nocedal, J., & Wright, S. J. (1999). *Numerical optimization*. Springer.

Ober, S. W., & Aitchison, L. (2021). Global inducing point variational posteriors for Bayesian neural networks and deep Gaussian processes. *International Conference on Machine Learning*.

Ober, S. W., Rasmussen, C. E., & van der Wilk, M. (2021). The promises and pitfalls of deep kernel learning. *Conference on Uncertainty in Artificial Intelligence*.

O'Hagan, A. (1978). Curve fitting and optimal design for prediction. *J. R. Stat. Soc., B: Stat. Methodol.*, **40**(1), 1–42.

O'Hagan, A. (1991). Bayes-Hermite quadrature. *J. Stat. Plan. Inference*, **29**(3), 245–260.

Øksendal, B. (2003). *Stochastic differential equations: An introduction with applications*. Springer.

Oseledets, I. V. (2011). Tensor-train decomposition. *SIAM J. Sci. Comput.*, **33**(5), 2295–2317.

Pascanu, R., Mikolov, T., & Bengio, Y. (2013). On the difficulty of training recurrent neural networks. *International Conference on Machine Learning*.

Pleiss, G., & Cunningham, J. P. (2021). The limitations of large width in neural networks: A deep Gaussian process perspective. *Advances in Neural Information Processing Systems*.

Popescu, S. G., Sharp, D. J., Cole, J. H., Kamnitsas, K., & Glocker, B. (2022). Distributional Gaussian processes layers for out-of-distribution detection. *Machine Learning for Biomedical Imaging*, **1**.

Prince, L. Y., Bakhtiari, S., Gillon, C. J., & Richards, B. A. (2021). Parallel inference of hierarchical latent dynamics in two-photon calcium imaging of neuronal populations. *bioRxiv:2021.03.05.434105, bioRxiv preprint*.

Quinonero-Candela, J., & Rasmussen, C. E. (2005). A unifying view of sparse approximate Gaussian process regression. *J. Mach. Learn. Res.*, **6**, 1939–1959.

Rasmussen, C. E., & Ghahramani, Z. (2001). Infinite mixtures of Gaussian process experts. *Advances in Neural Information Processing Systems*.

Rasmussen, C. E., & Williams, C. K. I. (2006). *Gaussian processes for machine learning*. MIT Press.

Reeb, D., Doerr, A., Gerwinn, S., & Rakitsch, B. (2018). Learning Gaussian processes by minimizing PAC-Bayesian generalization bounds. *Advances in Neural Information Processing Systems*.

Rezende, D. J., Mohamed, S., & Wierstra, D. (2014). Stochastic backpropagation and approximate inference in deep generative models. *International Conference on Machine Learning*.

Ripley, B. D. (1981). *Spatial statistics*. Wiley.

Ritter, H., Botev, A., & Barber, D. (2018). A scalable Laplace approximation for neural networks. *International Conference on Learning Representations*.

Rossi, S., Heinonen, M., Bonilla, E. V., Shen, Z., & Filippone, M. (2021). Sparse Gaussian processes revisited: Bayesian approaches to inducing-variable approximations. *International Conference on Artificial Intelligence and Statistics*.

Rudner, T. G. J., Sejdinovic, D., & Gal, Y. (2020). Inter-domain deep Gaussian processes. *International Conference on Machine Learning*.

Rue, H., Martino, S., & Chopin, N. (2009). Approximate Bayesian inference for latent Gaussian models by using integrated nested Laplace approximations. *J. R. Stat. Soc., B: Stat. Methodol.*, **71**(2), 319–392.

Ruttor, A., Batz, P., & Opper, M. (2013). Approximate Gaussian process inference for the drift function in stochastic differential equations. *Advances in Neural Information Processing Systems*.

Sacks, J., Welch, W. J., Mitchell, T. J., & Wynn, H. P. (1989). Design and analysis of computer experiments. *Stat. Sci.*, **4**(4), 409–423.

Salimbeni, H., & Deisenroth, M. P. (2017). Doubly stochastic variational inference for deep Gaussian processes. *Advances in Neural Information Processing Systems*.

Salimbeni, H., Eleftheriadis, S., & Hensman, J. (2018). Natural gradients in practice: Non-conjugate variational inference in Gaussian process models. *International Conference on Artificial Intelligence and Statistics*.

Salimbeni, H., Dutordoir, V., Hensman, J., & Deisenroth, M. P. (2019). Deep Gaussian processes with importance-weighted variational inference. *International Conference on Machine Learning*.

Salkuyeh, D. K. (2006). Comments on "a note on a three-term recurrence for a tridiagonal matrix". *Appl. Math. Comput.*, **176**(2), 442–444.

Särkkä, S. (2013). *Bayesian filtering and smoothing*. Cambridge University Press.

Särkkä, S., Solin, A., & Hartikainen, J. (2013). Spatiotemporal learning via infinite-dimensional Bayesian filtering and smoothing: A look at Gaussian process regression through kalman filtering. *IEEE Signal Process. Mag.*, **30**(4), 51–61.

Särkkä, S., & Solin, A. (2019). *Applied stochastic differential equations*. Cambridge University Press.

Schmidt, A. M., & O'Hagan, A. (2003). Bayesian inference for non-stationary spatial covariance structure via spatial deformations. *J. R. Statist. Soc. B*, **65**(3), 743–758.

Schön, T. B., & Lindsten, F. (2011). Manipulating the multivariate Gaussian density. *Technical Report, Linköping University*.

Schreiter, J., Nguyen-Tuong, D., Eberts, M., Bischoff, B., Markert, H., & Toussaint, M. (2015). Safe exploration for active learning with Gaussian processes. *European Conference on Machine Learning and Knowledge Discovery in Databases*.

Seeger, M. W., Williams, C. K. I., & Lawrence, N. D. (2003). Fast forward selection to speed up sparse Gaussian process regression. *International Workshop on Artificial Intelligence and Statistics*.

Settles, B. (2009). Active learning literature survey. *Technical Report, University of Wisconsin-Madison*.

Shahriari, B., Swersky, K., Wang, Z., Adams, R. P., & de Freitas, N. (2016). Taking the human out of the loop: A review of Bayesian optimization. *Proc. IEEE*, **104**(1), 148–175.

Skaug, H. J., & Fournier, D. A. (2006). Automatic approximation of the marginal likelihood in non-Gaussian hierarchical models. *Comput. Stat. Data Anal.*, **51**(2), 699–709.

Smola, A. J., & Bartlett, P. L. (2000). Sparse greedy Gaussian process regression. *Advances in Neural Information Processing Systems*.

Smolarski, D. C. (2006). Diagonally-striped matrices and approximate inverse preconditioners. *J. Comput. Appl. Math.*, **186**(2), 416–431.

Snelson, E. L., Rasmussen, C. E., & Ghahramani, Z. (2003). Warped Gaussian processes. *Advances in Neural Information Processing Systems*.

Snelson, E. L., & Ghahramani, Z. (2005). Sparse Gaussian processes using pseudo-inputs. *Advances in Neural Information Processing Systems*.

Snelson, E. L., & Ghahramani, Z. (2006). Variable noise and dimensionality reduction for sparse Gaussian processes. *Conference on Uncertainty in Artificial Intelligence*.

Snoek, J., Larochelle, H., & Adams, R. P. (2012). Practical Bayesian optimization of machine learning algorithms. *Advances in Neural Information Processing Systems*.

Snoek, J., Swersky, K., Zemel, R., & Adams, R. P. (2014). Input warping for Bayesian optimization of non-stationary functions. *International Conference on Machine Learning*.

Snoek, J., Ovadia, Y., Fertig, E., Lakshminarayanan, B., Nowozin, S., Sculley, D., Dillon, J. V., Ren, J., & Nado, Z. (2019). Can you trust your model's uncertainty? Evaluating predictive uncertainty under dataset shift. *Advances in Neural Information Processing Systems*.

Srinivas, N., Krause, A., Kakade, S. M., & Seeger, M. W. (2010). Gaussian process optimization in the bandit setting: No regret and experimental design. *International Conference on Machine Learning*.

Swiatkowski, J., Roth, K., Veeling, B. S., Tran, L., Dillon, J. V., Snoek, J., Mandt, S., Salimans, T., Jenatton, R., & Nowozin, S. (2020). The k-tied normal distribution: A compact parameterization of Gaussian mean field posteriors in Bayesian neural networks. *International Conference on Machine Learning*.

Thompson, P. D. (1956). Optimum smoothing of two-dimensional fields. *Tellus*, **8**(3), 384–393.

Titsias, M. (2009). Variational learning of inducing variables in sparse Gaussian processes. *International Conference on Artificial Intelligence and Statistics*.

Titsias, M., & Lázaro-Gredilla, M. (2014). Doubly stochastic variational Bayes for non-conjugate inference. *International Conference on Machine Learning*.

Tomczak, M., Swaroop, S., & Turner, R. (2020). Efficient low rank Gaussian variational inference for neural networks. *Advances in Neural Information Processing Systems*.

Tomczak, M., Swaroop, S., Foong, A. Y. K., & Turner, R. E. (2021). Collapsed variational bounds for Bayesian neural networks. *Advances in Neural Information Processing Systems*.

Toth, C., & Oberhauser, H. (2020). Bayesian learning from sequential data using Gaussian processes with signature covariances. *International Conference on Machine Learning*.

Toussaint, M. (2011). Gaussian identities. *Lecture Notes, University of Stuttgart*.

Tresp, V. (2000a). A Bayesian committee machine. *Neural Comput.*, **12**(11), 2719–2741.

Tresp, V. (2000b). Mixtures of Gaussian processes. *Advances in Neural Information Processing Systems*.

Turner, R. E., & Sahani, M. (2011). Two problems with variational expectation maximisation for time-series models. In *Bayesian time series models* (pp. 109–130). Cambridge University Press.

Tzen, B., & Raginsky, M. (2019). Neural stochastic differential equations: Deep latent Gaussian models in the diffusion limit. *arXiv preprint arXiv:1905.09883*.

Unlu, A., & Aitchison, L. (2021). Gradient regularization as approximate variational inference. *Entropy*, **23**(12), 1629.

Ustyuzhaninov, I., Kazlauskaite, I., Ek, C. H., & Campbell, N. (2020a). Monotonic Gaussian process flows. *International Conference on Artificial Intelligence and Statistics*.

Ustyuzhaninov, I., Kazlauskaite, I., Kaiser, M., Bodin, E., Campbell, N. D. F., & Ek, C. H. (2020b). Compositional uncertainty in deep Gaussian processes. *Conference on Uncertainty in Artificial Intelligence*.

van der Wilk, M., Rasmussen, C. E., & Hensman, J. (2017). Convolutional Gaussian processes. *Advances in Neural Information Processing Systems*.

van der Wilk, M. (2019). *Sparse Gaussian process approximations and applications* (Doctoral dissertation). University of Cambridge, UK.

Wang, J., Hertzmann, A., & Fleet, D. J. (2005). Gaussian process dynamical models. *Advances in Neural Information Processing Systems*.

Wang, K. A., Pleiss, G., Gardner, J. R., Tyree, S., Weinberger, K. Q., & Wilson, A. G. (2019). Exact Gaussian processes on a million data points. *Advances in Neural Information Processing Systems*.

Wenzel, F., Roth, K., Veeling, B. S., Swiatkowski, J., Tran, L., Mandt, S., Snoek, J., Salimans, T., Jenatton, R., & Nowozin, S. (2020a). How good is the Bayes posterior in deep neural networks really? *International Conference on Machine Learning*.

Wenzel, F., Snoek, J., Tran, D., & Jenatton, R. (2020b). Hyperparameter ensembles for robustness and uncertainty quantification. *Advances in Neural Information Processing Systems*.

Whittle, P. (1963). *Prediction and regulation by linear least-square methods*. English Universities Press.

Williams, C. K. I., & Rasmussen, C. E. (1995). Gaussian processes for regression. *Advances in Neural Information Processing Systems*.

Williams, R. J., & Zipser, D. (1995). Gradient-based learning algorithms for recurrent networks and their computational complexity. In *Backpropagation: Theory, architectures, and applications* (pp. 433–486). L. Erlbaum Associates Inc.

Williams, C. K. I., & Barber, D. (1998). Bayesian classification with Gaussian processes. *IEEE Trans. Pattern Anal. Mach. Intell.*, **20**(12), 1342–1351.

Williams, C. K. I., & Seeger, M. W. (2000). Using the Nyström method to speed up kernel machines. *Advances in Neural Information Processing Systems*.

Williams, C. K. I., Rasmussen, C. E., Schwaighofer, A., & Tresp, V. (2002). Observations on the Nyström method for Gaussian process prediction. *Technical Report, Max Planck Institute for Biological Cybernetics, Tübingen*.

Wilson, A. G., & Nickisch, H. (2015). Kernel interpolation for scalable structured Gaussian processes (KISS-GP). *International Conference on Machine Learning*.

Wilson, A. G., Hu, Z., Salakhutdinov, R., & Xing, E. P. (2016). Deep kernel learning. *International Conference on Artificial Intelligence and Statistics*.

Wilson, A. G., & Izmailov, P. (2020). Bayesian deep learning and a probabilistic perspective of generalization. *Advances in Neural Information Processing Systems*.

Wilson, J., Borovitskiy, V., Terenin, A., Mostowsky, P., & Deisenroth, M. P. (2020). Efficiently sampling functions from Gaussian process posteriors. *International Conference on Machine Learning*.

Yıldız, Ç., Heinonen, M., Intosalmi, J., Mannerström, H., & Lähdesmäki, H. (2018). Learning stochastic differential equations with Gaussian processes without gradient matching. *International Workshop on Machine Learning for Signal Processing*.

Yıldız, Ç., Heinonen, M., & Lahdesmäki, H. (2019). ODE2VAE: Deep generative second order ODEs with Bayesian neural networks. *Advances in Neural Information Processing Systems*.

Yu, H., Chen, Y., Low, B. K. H., Jaillet, P., & Dai, Z. (2019). Implicit posterior variational inference for deep Gaussian processes. *Advances in Neural Information Processing Systems*.

Yu, C., Seslija, M., Brownbridge, G., Mosbach, S., Kraft, M., Parsi, M., Davis, M., Page, V., & Bhave, A. (2020). Deep kernel learning approach to engine emissions modeling. *Data-Centric Engineering*, **1**.

Yuksel, S. E., Wilson, J. N., & Gader, P. D. (2012). Twenty years of mixture of experts. *IEEE Trans. Neural Netw. Learn. Syst.*, **23**(8), 1177–1193.

Zhao, Z., Tronarp, F., Hostettler, R., & Särkkä, S. (2020). State-space Gaussian process for drift estimation in stochastic differential equations. *International Conference on Acoustics, Speech and Signal Processing*.

Zimmer, C., Meister, M., & Nguyen-Tuong, D. (2018). Safe active learning for time-series modeling with Gaussian processes. *Advances in Neural Information Processing Systems*.

Zimmer, C., & Yaesoubi, R. (2020). Influenza forecasting framework based on Gaussian processes. *International Conference on Machine Learning*.