Bayesian Time Series Learning with Gaussian Processes

Roger Frigola-Alcalde
Department of Engineering
St Edmund's College
University of Cambridge

August 2015

This dissertation is submitted for the degree of
Doctor of Philosophy

SUMMARY

The analysis of time series data is important in fields as disparate as the social sciences, biology, engineering or econometrics. In this dissertation, we present a number of algorithms designed to learn Bayesian nonparametric models of time series. The goal of these kinds of models is twofold. First, they aim at making predictions which quantify the uncertainty due to limitations in the quantity and the quality of the data. Second, they are flexible enough to model highly complex data whilst preventing overfitting when the data does not warrant complex models.

We begin with a unifying literature review on time series models based on Gaussian processes. Then, we centre our attention on the Gaussian Process State-Space Model (GP-SSM): a Bayesian nonparametric generalisation of discrete-time nonlinear state-space models. We present a novel formulation of the GP-SSM that offers new insights into its properties. We then proceed to exploit those insights by developing new learning algorithms for the GP-SSM based on particle Markov chain Monte Carlo and variational inference.

Finally, we present a filtered nonlinear auto-regressive model with a simple, robust and fast learning algorithm that makes it well suited to its application by non-experts on large datasets. Its main advantage is that it avoids the computationally expensive (and potentially difficult to tune) smoothing step that is a key part of learning nonlinear state-space models.

Acknowledgements

I would like to thank Carl Rasmussen who took the gamble of accepting a candidate for a PhD in statistical machine learning who barely knew what a probability distribution was. I am also grateful to Zoubin Ghahramani, Rich Turner and Daniel Wolpert for creating such a wonderful academic environment at the Computational and Biological Learning Lab in Cambridge; it has been incredible to interact with so many talented people. I would also like to thank Thomas Schön for hosting my fruitful academic visit at Linköping University (Sweden).

I am indebted to my talented collaborators Fredrik Lindsten and Yutian Chen. Their technical skills have been instrumental to develop ideas that were only fuzzy in my head.

I would also like to thank my thesis examiners, Rich Turner and James Hensman, and Thang Bui for their helpful feedback that has certainly improved the thesis. Of course, any remaining inaccuracies and errors are entirely my fault!

Finally, I can not thank enough all the colleagues, friends and family with whom I have spent so many joyful times throughout my life.

Declaration

This dissertation is the result of my own work and includes nothing which is the outcome of work done in collaboration except as specified in the text.

This dissertation is not substantially the same as any that I have submitted, or, is being concurrently submitted for a degree or diploma or other qualification at the University of Cambridge or any other University or similar institution. I further state that no substantial part of my dissertation has already been submitted, or, is being concurrently submitted for any such degree, diploma or other qualification at the University of Cambridge or any other University of similar institution.

This dissertation does not exceed 65,000 words, including appendices, bibliography, footnotes, tables and equations. This dissertation does not contain more than 150 figures.

"Would you tell me, please, which way I ought to go from here?"

"That depends a good deal on where you want to get to," said the Cat.

"I don't much care where —" said Alice.

"Then it doesn't matter which way you go," said the Cat.

"— so long as I get *somewhere*," Alice added as an explanation.

"Oh, you're sure to do that," said the Cat, "if you only walk long enough."

Lewis Carroll, *Alice's Adventures in Wonderland*

Contents

Chapter 1

Introduction

"The purpose of computing is insight, not numbers."

Richard W. Hamming

"The purpose of computing is numbers — specifically, correct numbers."

Leslie F. Greengard

1.1 Time Series Models

Time series data consists of a number of measurements taken over time. For example, a time series dataset could be created by recording power generated by a solar panel, by storing measurements made by sensors on an aircraft, or by monitoring the vital signs of a patient in a hospital. The ubiquity of time series data makes its analysis important for fields as disparate as the social sciences, biology, engineering or econometrics.

Time series tend to exhibit high correlations induced by the temporal structure in the data. It is therefore not surprising that specialised methods for time series analysis have been developed over time. In this thesis we will focus on a model-based approach to time series analysis. Models are mathematical constructions that often correspond to an idealised view about how the data is generated. Models are useful to make predictions about the future and to better understand what was happening while the data was being recorded.

The process of tuning models of time series using data is called system identification in the field of control theory (Ljung, 1999). In the fields of statistics and machine learning it is often referred to as estimation, fitting, inference or learning of time series (Hamilton, 1994; Shumway and Stoffer, 2011; Barber et al., 2011; Tsay, 2013). Creating faithful models given the data at our disposal is of great practical importance. Otherwise any reasoning, prediction or design based on the data could be fatally flawed.

Models are usually not perfect and the data available to us is often limited in quantity and quality. It is therefore normal to ask ourselves: are we limited

to creating just one model or could we make a collection of models that were all plausible given the available data? If we embrace uncertainty, we can move from the notion of having a single model to that of keeping a potentially infinite collection of models and combining them to make decisions of any sort. This is one of the fundamental ideas behind Bayesian inference.

In this thesis, we develop methods for Bayesian inference applied to dynamical systems using models based on Gaussian processes. Although we will work with very *general* models that can be applied in a variety of situations, our mindset is that of the field of system identification. In other words, we focus on learning models typically found in engineering problems where a relatively limited amount of noisy sensors give a glimpse at the complex dynamics of a system.

1.2 BAYESIAN NONPARAMETRIC TIME SERIES MODELS

1.2.1 BAYESIAN METHODS

When learning a model from a time series, we never have the luxury of an infinite amount of noiseless data and unlimited computational power. In practice, we deal with finite noisy datasets which lead to uncertainty about what the most appropriate model is given the available data. In Bayesian inference, probabilities are treated as a way to represent the subjective uncertainty of the rational agent performing inference (Jaynes, 2003). This uncertainty is represented as a probability distribution over the model given the data

$$p(\mathcal{M} \mid \mathcal{D}), \tag{1.1}$$

where the model is understood here in the sense of the functional form *and* the value of any parameter that the model might have. This contrasts with the most common approaches to time series analysis where a *single* model of the system is found, usually by optimising a cost function such as the likelihood (Ljung, 1999; Shumway and Stoffer, 2011). After optimisation, the resulting model is considered *the* best available representation of the system and used for any further application.

In a Bayesian approach, however, it is acknowledged that several models (or values of a parameter) can be consistent with the data (Peterka, 1981). In the case of a parametric model, rather than obtaining a single estimate of the "best" value of the parameters θ^*, Bayesian inference will produce a posterior probability distribution over this parameter $p(\theta|\mathcal{D})$. This distribution allows for the fact that several values of the parameter might also be plausible given the observed data $\mathcal{D}$. The posterior $p(\theta|\mathcal{D})$ can then interpreted as our degree of belief about the value of the parameter θ. Predictions or decisions based on the posterior are made by computing expectations over the posterior. Informally, one can think of predictions as being an average using different values

of the parameters weighted by how much the parameter is consistent with the data. Predictions can be made with error bars that represent both the system's inherent stochasticity *and* our own degree of ignorance about what the correct model is.

For the sake of example, let's consider a parametric model of a discrete-time stochastic dynamical system with a continuous state defined by x_t. The state transition density is

$$p(x_{t+1}|x_t, \theta). \tag{1.2}$$

Bayesian learning provides a posterior over the unknown parameter θ given the data $p(\theta|\mathcal{D})$. To make predictions about the state transition we can integrate over the posterior in order to average over all plausible values of the parameter after having seen the data

$$p(x_{t+1}|x_t, \mathcal{D}) = \int p(x_{t+1}|x_t, \theta)\, p(\theta|\mathcal{D})\, d\theta. \tag{1.3}$$

Here, predictions are made by considering all plausible values of θ, not only a "best guess".

Bayesian inference needs a prior: $p(\theta)$ for our example above. The prior is a probability distribution representing our uncertainty about the object to be inferred *before* seeing the data. This requirement of a subjective distribution is often criticised. However, the prior is an opportunity to formalise many of the assumptions that in other methods may be less explicit. MacKay (2003) points out that "*you cannot do inference without making assumptions*". In Bayesian inference those assumptions are very clearly specified.

In the context of learning dynamical systems, Isermann and Münchhof (2011) derive maximum likelihood and least squares methods from Bayesian inference. Although maximum likelihood and Bayesian inference are related in their use of probability, there is a fundamental philosophical difference. Spiegelhalter and Rice (2009) summarise it eloquently:

> *At a simple level, 'classical' likelihood-based inference closely resembles Bayesian inference using a flat prior, making the posterior and likelihood proportional. However, this underestimates the deep philosophical differences between Bayesian and frequentist inference; Bayesian* [sic] *make statements about the relative evidence for parameter values given a dataset, while frequentists compare the relative chance of datasets given a parameter value.*

Bayesian methods are also relevant in "big data" settings when the complexity of the system that generated the data is large relative to the amount of data. Large datasets can contain many nuances of the behaviour of a complex system. Models of large capacity are then required if those nuances are to be captured properly. Moreover, there will still be uncertainty about the system that generated the data. For example, even if a season's worth of time series data recorded from a Formula 1 car is in the order of a petabyte, there will be, hopefully, very little data of the car sliding sideways out of control. Therefore,

models about the behaviour of the car at extremely large slip angles would be highly uncertain.

1.2.2 BAYESIAN METHODS IN SYSTEM IDENTIFICATION

Despite Peterka (1981) having set the basis for Bayesian system identification, Bayesian system identification has not had a significant impact within the control community. For instance, consider this quote from a recent book by well known authors in the field (Isermann and Münchhof, 2011):

> *Bayes estimation has little relevance for practical applications in the area of system identification.* [...] *Bayes estimation is mainly of theoretical value. It can be regarded as the most general and most comprehensive estimation method. Other fundamental estimation methods can be derived from this starting point by making certain assumptions or specializations.*

It is apparent that the authors recognise Bayesian inference as a sound framework for estimation but later dismiss it on the grounds of its mathematical and computational burden and its need for prior information.

However, since Peterka's article in 1981, there have been radical improvements in both computational power and algorithms for Bayesian learning. Some influential recent developments such as the regularised kernel methods of Chen et al. (2012) can be interpreted from a Bayesian perspective. Their use of a prior for regularised learning has shown great promise and moving from this to the creation of a posterior over dynamical systems is straightforward.

1.2.3 NONPARAMETRIC MODELS

In Equation (1.3) we have made use of the fact that

$$p(x_{t+1}|x_t, \theta, \mathcal{D}) = p(x_{t+1}|x_t, \theta), \tag{1.4}$$

which is true for parametric models: predictions are conditionally independent of the observed data $\mathcal{D}$ given the parameters. In other words, the data is distilled into the parameter θ and any subsequent prediction does not make use of the original dataset. This is very convenient but it is not without its drawbacks. Choosing a model from a particular parametric class constrains its flexibility. An alternative is to use *nonparametric models*. In those models, the data is not reduced to a finite set of parameters. In fact, nonparametric models can be shown to have an infinite-dimensional parameter space (Orbanz, 2014). This allows the model to represent more complexity as the size of the dataset $\mathcal{D}$ grows; defying in this way the bound in model complexity existing in parametric models.

Models can be thought of as an information channel from past data to future predictions (Ghahramani, 2012). In this context, a parametric model constitutes a bottleneck in the information channel: predictions are made based only on the learnt parameters. However, nonparametric models are *memory-based* since

they need to "remember" the full dataset in order to make predictions. This can be interpreted as nonparametric models having a number of parameters that progressively grows with the size of the dataset.

Bayesian nonparametric models combine the two aspects presented up to this point: they allow Bayesian inference to be performed on objects of infinite dimensionality. Next chapter introduces how Gaussian processes (Rasmussen and Williams, 2006) can be used to infer a function given data. The result of inference will not be a single function but a posterior distribution over functions.

1.3 CONTRIBUTIONS

The main contributions in this dissertation have been published before in

- Roger Frigola and Carl E. Rasmussen, (2013). Integrated preprocessing for Bayesian nonlinear system identification with Gaussian processes. In *IEEE 52nd Annual Conference on Decision and Control (CDC)*, pp. 5371-5376.
- Roger Frigola, Fredrik Lindsten, Thomas B. Schön, and Carl E. Rasmussen. (2013). Bayesian inference and learning in Gaussian process state-space models with particle MCMC. In *Advances in Neural Information Processing Systems 26 (NIPS)*, pp. 3156-3164.
- Roger Frigola, Fredrik Lindsten, Thomas B. Schön, and Carl E. Rasmussen. (2014). Identification of Gaussian process state-space models with particle stochastic approximation EM. In *19th World Congress of the International Federation of Automatic Control (IFAC)*, pp. 4097-4102.
- Roger Frigola, Yutian Chen, and Carl E. Rasmussen. (2014). Variational Gaussian process state-space models. In *Advances in Neural Information Processing Systems 27 (NIPS)*, pp. 3680-3688.

However, I take the opportunity that the thesis format offers to build on the publications and extend the presentation of the material. In particular, the lack of asphyxiating page count constraints will allow for the presentation of subtle details of Gaussian Process State-Space Models that could not fit in the papers. Also, there is now space for a comprehensive review highlighting prior work using Gaussian processes for time series modelling. Models that were originally presented in different contexts and with differing notations are now put under the same light for ease of comparison. Hopefully, this review will be useful to researchers entering the field of time series modelling with Gaussian processes.

A summary of technical contributions of this thesis can be found in Section 7.1.

Chapter 2

Time Series Modelling with Gaussian Processes

This chapter aims at providing a self-contained review of previous work on time series modelling with Gaussian processes. A particular effort has been made to present models with a unified notation and separate the models themselves from the algorithms used to learn them.

2.1 INTRODUCTION

Learning dynamical systems, also known as system identification or time series modelling, aims at the creation of a model based on measured signals. This model can be used, amongst other things, to predict future behaviour of the system, to explain interesting structure in the data or to denoise the original time series.

Systems displaying temporal dynamics are ubiquitous in nature, engineering and the social sciences. For instance, we could obtain data about how the number of individuals of several species in an ecosystem changes with time; we could record data from sensors in an aircraft; or we could log the evolution of share price in a stock market. In all those cases, learning from time series can provide both insight and the ability to make predictions.

We consider that there will potentially be two kinds of measured signals. In the system identification jargon those signals are named inputs and outputs. Inputs are external influences to the system (e.g. rain in an ecosystem or turbulence affecting an aircraft) and outputs are signals that depend on the current properties of the system (e.g. the number of lions in a savannah or the speed of an aircraft). We will denote the input vector at time t by $\mathbf{u}_t \in \mathbb{R}^{n_u}$ and the output vector by $\mathbf{y}_t \in \mathbb{R}^{n_y}$.

In this review we are mainly concerned with two important families of dynamical system models (Ljung, 1999). First, auto-regressive (AR) models directly model the next output of a system as a function of a number of previous

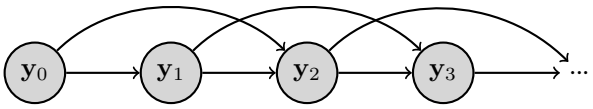


Figure 2.1: Second order ($\tau_y = 2$) auto-regressive model.

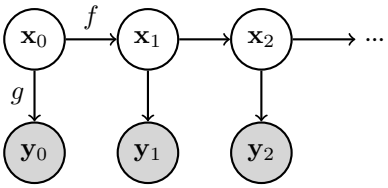


Figure 2.2: State-space model. Shaded nodes are observed and unshaded nodes are latent (hidden).

inputs and outputs

$$\mathbf{y}_t = f(\mathbf{y}_{t-1}, ..., \mathbf{y}_{t-\tau_y}, \mathbf{u}_{t-1}, ..., \mathbf{u}_{t-\tau_u}) + \boldsymbol{\delta}_t, \tag{2.1}$$

where $\boldsymbol{\delta}_t$ represents random noise that is independent and identically distributed across time.

The second class of dynamical system models, named state-space models (SSM), introduces latent (unobserved) variables called states $\mathbf{x}_t \in \mathbb{R}^{n_x}$. The state at a given time summarises all the history of the system and is enough to make predictions about its future. A state-space model is mainly defined by the state transition function f and the measurement function g

$$\mathbf{x}_{t+1} = f(\mathbf{x}_t, \mathbf{u}_t) + \mathbf{v}_t, \tag{2.2a}$$

$$\mathbf{y}_t = g(\mathbf{x}_t, \mathbf{u}_t) + \mathbf{e}_t, \tag{2.2b}$$

where $\mathbf{v}_t$ and $\mathbf{e}_t$ are additive noises known as the process noise and measurement noise, respectively.

From this point on, in the interest of notational simplicity, we will avoid explicitly conditioning on observed inputs. When available, inputs can always be added as arguments to the various functions that are being learnt. Figure 2.1 represents the graphical model of an auto-regressive model and Figure 2.2 is the graphical model of a state-space model.

2.2 GAUSSIAN PROCESSES

Gaussian processes (GPs) are a class of stochastic processes that have proved very successful to perform inference directly over the space of functions (Rasmussen and Williams, 2006). This contrasts with models of functions defined

by a parameterised class of functions and a prior over the parameters. In the context of modelling dynamical systems, Gaussian processes can be used as priors over the functions representing the system dynamics or the mapping from latent states to measurements.

In the following, we provide a brief exposition of Gaussian processes and their application to statistical regression problems. We refer the reader to (Rasmussen and Williams, 2006) for a detailed description. A Gaussian process can be defined as a collection of random variables, any *finite* number of which have a joint Gaussian distribution

$$p(\mathbf{f}_i, \mathbf{f}_j, \mathbf{f}_k, ...) = \mathcal{N}\left(\begin{bmatrix} m(\mathbf{x}_i) \\ m(\mathbf{x}_j) \\ m(\mathbf{x}_k) \\ \vdots \end{bmatrix}, \begin{bmatrix} k(\mathbf{x}_i, \mathbf{x}_i) & k(\mathbf{x}_i, \mathbf{x}_j) & k(\mathbf{x}_i, \mathbf{x}_k) & \\ k(\mathbf{x}_j, \mathbf{x}_i) & k(\mathbf{x}_j, \mathbf{x}_j) & k(\mathbf{x}_j, \mathbf{x}_k) & \\ k(\mathbf{x}_k, \mathbf{x}_i) & k(\mathbf{x}_k, \mathbf{x}_j) & k(\mathbf{x}_k, \mathbf{x}_k) & \\ & & & \ddots \end{bmatrix}\right). \tag{2.3}$$

The value of a function at a particular input location $f(\mathbf{x}_i)$ is denoted by the random variable $\mathbf{f}_i$. To denote that a function follows a Gaussian process, we write

$$f(\mathbf{x}) \sim \mathcal{GP}\left(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}')\right), \tag{2.4}$$

where $m(\mathbf{x})$ and $k(\mathbf{x}, \mathbf{x}')$ are the mean and covariance functions respectively. Those two functions fully specify the Gaussian process.

2.2.1 GAUSSIAN PROCESSES FOR REGRESSION

The regression problem is perhaps the simplest in which one can appreciate the usefulness of Gaussian processes for machine learning. The task consists in learning from a dataset with input-output data pairs $\{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^N$ where the outputs are real-valued. After learning, it is possible to predict the value of the output $\mathbf{y}_*$ at any new test input $\mathbf{x}_*$. Regression consists, therefore, in learning the function mapping inputs to outputs: $\mathbf{y}_* = f(\mathbf{x}_*)$. In the following, we consider how to perform Bayesian inference in the space of functions with the help of Gaussian processes.

When doing Bayesian inference on a parametric model we put a prior on the parameter of interest $p(\theta)$ and obtain a posterior distribution over the parameter given the data by combining the prior with the likelihood function $p(y|\theta)$:

$$p(\theta|y) = \frac{p(y|\theta)\, p(\theta)}{p(y)}, \tag{2.5}$$

where $p(y)$ is called the *evidence* or the *marginal likelihood* and depends on the prior and the likelihood

$$p(y) = \int p(y, \theta)\, d\theta = \int p(y|\theta)\, p(\theta)\, d\theta. \tag{2.6}$$

In regression, the ultimate goal is to infer the *function* mapping inputs to outputs. A parametric approach to Bayesian regression consists in specifying

a family of functions parameterised by a finite set of parameters, putting a prior on those parameters and performing inference. However, we can find a less restrictive and very powerful approach to inference on functions by directly specifying a prior over an infinite-dimensional space of functions. This contrasts with putting a prior over a finite set of parameters which implicitly specify a distribution over functions.

A very useful prior over functions is the Gaussian process. In a Gaussian process, once we have selected a finite collection of points $\mathbf{x} \triangleq \{\mathbf{x}_1, ..., \mathbf{x}_N\}$ at which to evaluate a function, the prior distribution over the values of the function at those locations, $\mathbf{f} \triangleq \big(f(\mathbf{x}_1), ..., f(\mathbf{x}_N)\big)$, is a Gaussian distribution

$$p(\mathbf{f}|\mathbf{x}) = \mathcal{N}(\mathbf{m}(\mathbf{x}), \mathbf{K}(\mathbf{x})), \tag{2.7}$$

where $\mathbf{m}(\mathbf{x})$ and $\mathbf{K}(\mathbf{x})$ are the mean vector and covariance matrix defined in the same way as in Equation (2.3). This is due to the marginal of a Gaussian process being a Gaussian distribution. Therefore, when we only deal with the Gaussian process at a finite set of inputs, computations involving the prior are based on Gaussian distributions. Bayes' theorem can be applied in the conventional manner to obtain a posterior over the latent function at all locations $\mathbf{x}$ where observations $\mathbf{y} \triangleq \{\mathbf{y}_1, ..., \mathbf{y}_N\}$ are available

$$p(\mathbf{f}|\mathbf{y}, \mathbf{x}) = \frac{p(\mathbf{y}|\mathbf{f})\, p(\mathbf{f}|\mathbf{x})}{p(\mathbf{y}|\mathbf{x})} = \frac{p(\mathbf{y}|\mathbf{f})\, \mathcal{N}(\mathbf{f}|\mathbf{m}(\mathbf{x}), \mathbf{K}(\mathbf{x}))}{p(\mathbf{y}|\mathbf{x})}. \tag{2.8}$$

And since the denominator is a constant for any given dataset[1], we note the proportionality

$$p(\mathbf{f}|\mathbf{y}, \mathbf{x}) \propto p(\mathbf{y}|\mathbf{f})\, \mathcal{N}(\mathbf{f}|\mathbf{m}(\mathbf{x}), \mathbf{K}(\mathbf{x})). \tag{2.9}$$

In the *particular case* where the likelihood has the form $p(\mathbf{y}|\mathbf{f}) = \mathcal{N}(\mathbf{y}|\mathbf{f}, \mathbf{\Sigma}_n)$, this posterior can be computed analytically and is Gaussian

$$\begin{aligned} p(\mathbf{f}|\mathbf{y}, \mathbf{x}) = \mathcal{N}(\mathbf{f}|&\mathbf{K}(\mathbf{x})\big(\mathbf{K}(\mathbf{x}) + \mathbf{\Sigma}_n\big)^{-1}\big(\mathbf{y} - \mathbf{m}(\mathbf{x})\big), \\ &\mathbf{K}(\mathbf{x}) - \mathbf{K}(\mathbf{x})\big(\mathbf{K}(\mathbf{x}) + \mathbf{\Sigma}_n\big)^{-1}\mathbf{K}(\mathbf{x})). \end{aligned} \tag{2.10}$$

This is the case in Gaussian process regression when additive Gaussian noise is considered. However, for arbitrary likelihood functions the posterior will not necessarily be Gaussian.

The distribution $p(\mathbf{f}|\mathbf{y}, \mathbf{x})$ represents the posterior over the latent function $f(\mathbf{x})$ at all locations in the set $\mathbf{x}$. This can be useful in itself, but we may also be interested in the value of $f(\mathbf{x})$ at other locations in the input space. In other words, we may be interested in the *predictive* distribution of $\mathbf{f}_* = f(\mathbf{x}_*)$ at a new location $\mathbf{x}_*$

$$p(\mathbf{f}_*|\mathbf{x}_*, \mathbf{y}, \mathbf{x}). \tag{2.11}$$

[1] Note that we are currently considering mean and covariance functions that do not have hyperparameters to be tuned.

This can be achieved by marginalising $\mathbf{f}$

$$p(\mathbf{f}_*|\mathbf{x}_*, \mathbf{y}, \mathbf{x}) = \int p(\mathbf{f}_*, \mathbf{f}|\mathbf{x}_*, \mathbf{y}, \mathbf{x})\, d\mathbf{f} = \int p(\mathbf{f}_*|\mathbf{x}_*, \mathbf{f}, \mathbf{x})\, p(\mathbf{f}|\mathbf{y}, \mathbf{x})\, d\mathbf{f}, \quad (2.12)$$

where the first term in the second integral is always a Gaussian that results from the Gaussian process prior linking all possible values of $\mathbf{f}$ and $\mathbf{f}_*$ with a joint normal distribution (Rasmussen and Williams, 2006). The second term, $p(\mathbf{f}|\mathbf{y}, \mathbf{x})$, is simply the posterior of $\mathbf{f}$ from Equation (2.8). When the likelihood is $p(\mathbf{y}|\mathbf{f}) = \mathcal{N}(\mathbf{y}|\mathbf{f}, \mathbf{\Sigma}_n)$, both the posterior and predictive distributions are Gaussian. For other likelihoods one may need to resort to approximation methods (e.g. (Murray et al., 2010; Nguyen and Bonilla, 2014)).

For Gaussian likelihoods it is also straightforward to marginalise the unknown values of the function and obtain a tractable marginal likelihood of the model

$$p(\mathbf{y}|\mathbf{x}) = \mathcal{N}(\mathbf{y}|\mathbf{m}(\mathbf{x}), \mathbf{K}(\mathbf{x}) + \mathbf{\Sigma}_n). \quad (2.13)$$

Maximising the marginal likelihood with respect to the mean and covariance functions provides a practical way to perform Bayesian model selection (MacKay, 2003; Rasmussen and Williams, 2006).

2.2.2 GRAPHICAL MODELS OF GAUSSIAN PROCESSES

A possible way to represent Gaussian processes in a graphical model consists in drawing a thick solid bar between the jointly normally-distributed variables (Rasmussen and Williams, 2006). All the variables that touch the solid bar belong to the same Gaussian process and are fully interconnected, i.e. in principle no conditional independence statements between those variables can be made until one examines the covariance function. There are an infinite amount of variables in the Gaussian process but we only draw a finite set. Periods of ellipsis can be drawn at the extremities of the solid bar to reinforce this idea.

Figure 2.3 depicts the model for Gaussian process regression with a dataset consisting of three inputs and three outputs and where predictions are to be made at a given test point $\mathbf{x}_*$.

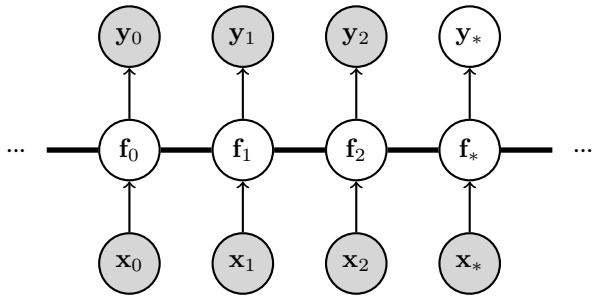


Figure 2.3: Graphical model of Gaussian process regression.

A way to interpret this notation consists in considering a function $f(\cdot)$ that is distributed according to a GP. All variables $\mathbf{f}$ are conditionally independent of each other given that function (Figure 2.4).

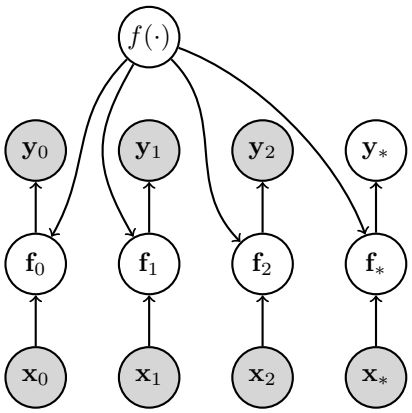


Figure 2.4: Graphical model of Gaussian process regression explicitly representing the latent function $f(\cdot)$.

It is important to note that models involving the thick bar should *not* be interpreted as a conventional undirected probabilistic graphical model. The thick bar notation shows in a simple manner which variables belong to the same GP. For general covariance functions, drawing Figure 2.3 with the semantics of conventional directed graphical models would result in a large number of edges. See Figure 2.5 for an example.

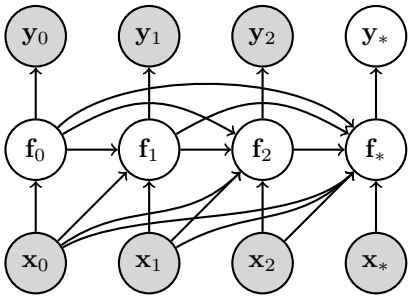


Figure 2.5: Graphical model of Gaussian process regression using only directed edges.

2.3 A ZOO OF GP-BASED DYNAMICAL SYSTEM MODELS

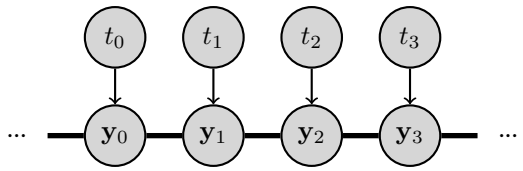
In this section we present a unified description of several models of dynamical systems based on Gaussian process that have appeared in the literature. In particular, we provide generative models to easily identify the assumptions made by each model and review the different inference/learning algorithms that have been tailored to each model.

The models based on Gaussian processes presented in this section are in fact generalisations of parametric models with similar structures. For instance, if the covariance functions in the GPs of the model shown in Section 2.3.5 are selected to be linear, the model becomes equivalent to the usual linear (parametric) state-space model.

Furthermore, the equivalence theorem presented in Section 3.2.1 implies that several GP-based state-space models that have originally been presented as different in the literature are in fact equivalent. Those equivalences will be highlighted in the description of each model.

2.3.1 LINEAR-GAUSSIAN TIME SERIES MODEL

Graphical model:



Generative model:

$$\mathbf{y}(t) \sim \mathcal{GP}\big(m(t), k(t,t')\big). \tag{2.14}$$

Description: Many popular linear-Gaussian time series models can be interpreted as Gaussian processes with a particular covariance function and time as the index set of the GP. This includes linear auto-regressive models, linear auto-regressive moving-average models, linear state-space models[2], etc. A characteristic of these models is that all observations $\mathbf{y}_{1:T}$ are *jointly Gaussian*. Linear-Gaussian time series models have been studied in the classical time series literature (Box et al., 1994), stochastic processes literature (Grimmett and Stirzaker, 2001) and also from a Gaussian process perspective (Turner, 2011; Roberts et al., 2012).

Inference and Learning: There is a rich literature about learning models of various flavours of linear auto-regressive models. Both maximum likelihood and Bayesian learning approaches have been described (Box et al., 1994; Ljung, 1999).

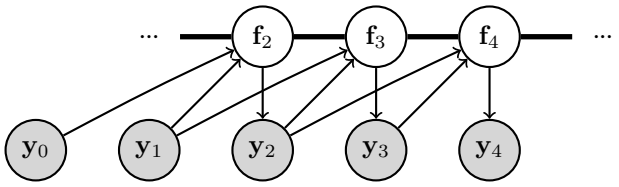
Inference in linear-Gaussian state-space models can be performed efficiently with the Kalman filter/smoother. Their use leads to $\mathcal{O}(T)$ exact inference rather than the naive $\mathcal{O}(T^3)$. Learning in linear-Gaussian state-space models has been tackled with approaches such as maximum likelihood (Shumway and Stoffer, 1982; Ljung, 1999), subspace methods (a type of spectral learning) (Overschee and Moor, 1996), and variational Bayes (Barber and Chiappa, 2006).

A different approach is to treat the problem as Gaussian process regression with common covariance functions such as the squared exponential which specifies correlations that decay with time difference or periodic covariance functions that impose a period to the signal. See, for instance, (Roberts et al., 2012) for a recent overview of this approach and (Duvenaud et al., 2013) for an algorithm to perform search in the space of GP kernels. It is important to note, however, that GP regression from time to observations has an important drawback: it is not able to learn nonlinear dynamics. For example, consider a nonlinear aerobatic aeroplane. A model such as the one in this section can be useful to filter or interpolate data from a given test flight. However, it will not be able to learn a model of the aeroplane nonlinear dynamics suitable to create a flight simulator.

[2]Sometimes known as Kalman filters although this name can lead to confusion between the model and the algorithm that is used to perform efficient exact inference on it.

2.3.2 NONLINEAR AUTO-REGRESSIVE MODEL WITH GP

Graphical model:



Second order model, i.e. $\mathbf{Y}_{t-1} = \{\mathbf{y}_{t-1}, \mathbf{y}_{t-2}\}$.

Generative model:

$$f(\mathbf{Y}) \sim \mathcal{GP}\big(m_f(\mathbf{Y}), k_f(\mathbf{Y}, \mathbf{Y}')\big), \tag{2.15a}$$

$$\mathbf{Y}_{\tau_y - 1} \sim p(\mathbf{Y}_{\tau_y - 1}), \tag{2.15b}$$

$$\mathbf{f}_t = f(\mathbf{Y}_{t-1}), \tag{2.15c}$$

$$\mathbf{y}_t \mid \mathbf{f}_t \sim p(\mathbf{y}_t \mid \mathbf{f}_t, \boldsymbol{\theta}), \tag{2.15d}$$

where

$$\mathbf{Y}_{t-1} = \{\mathbf{y}_{t-1}, ..., \mathbf{y}_{t-\tau_y}\}. \tag{2.15e}$$

Description: Auto-regressive models describe a time series by defining a mapping from past observations to the current observation. In the case of additive noise this is equivalent to

$$\mathbf{y}_t = f(\mathbf{y}_{t-1}, ..., \mathbf{y}_{t-\tau_y}) + \boldsymbol{\delta}_t.$$

Conceptually, in other to generate data from this model, one would initially draw a function from Equation (2.15a) and draw the first τ_y observations (Eq. (2.15b)). Then,the rest of the variables could be drawn sequentially. In practice, however, it is not generally possible to sample a whole function from the GP since it is an infinite dimensional object. Confer to section 3.1 for a discussion on how to draw samples in practice in a related model.

An important characteristic of this model is that there is no measurement noise such as that in a state-space model. Here, noise injected via $\boldsymbol{\delta}_t$ has an influence on the *future trajectory* of the system.

Nonlinear auto-regressive models with external (exogenous) inputs are often known as NARX models.

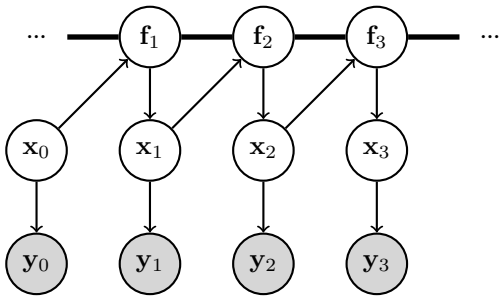
Inference and Learning: Learning in this model is performed using conventional GP regression techniques. Given a Gaussian process prior on the latent function, all flavours of Gaussian process regression can be applied to this model. In particular, exact inference is possible if we choose a conjugate likelihood, e.g. $p(\mathbf{y}_t \mid \mathbf{f}_t, \boldsymbol{\theta}) = \mathcal{N}(\mathbf{y}_t \mid \mathbf{f}_t, \mathbf{R})$.

Gregorcic and Lightbody (2002) and Kocijan et al. (2003) presented learning of a GP-based NARX model via maximisation of the marginal likelihood. Girard et al. (2003) proposed a method to propagate the predictive uncertainty

in GP-based NARX models for multiple-step ahead forecasting. More recently, Gutjahr et al. (2012) have proposed the use of sparse Gaussian processes in order to reduce computational complexity and to scale to time series with millions of data points.

2.3.3 STATE-SPACE MODEL WITH TRANSITION GP

Graphical model:



Generative model:

$$f(\mathbf{x}) \sim \mathcal{GP}\big(m_f(\mathbf{x}), k_f(\mathbf{x}, \mathbf{x}')\big), \tag{2.16a}$$

$$\mathbf{x}_0 \sim p(\mathbf{x}_0), \tag{2.16b}$$

$$\mathbf{f}_t = f(\mathbf{x}_{t-1}), \tag{2.16c}$$

$$\mathbf{x}_t \mid \mathbf{f}_t \sim \mathcal{N}(\mathbf{f}_t, \mathbf{Q}), \tag{2.16d}$$

$$\mathbf{y}_t \mid \mathbf{x}_t \sim p(\mathbf{y}_t \mid \mathbf{x}_t, \boldsymbol{\theta}_y). \tag{2.16e}$$

Description: This model corresponds to a nonlinear state-space model where a Gaussian process prior has been placed over the state transition function $f(\mathbf{x}_t)$. This model can be seen as a generalisation of the parametric state-space model described by

$$\mathbf{x}_{t+1} \mid \mathbf{x}_t \sim \mathcal{N}(\tilde{f}(\mathbf{x}_t, \boldsymbol{\theta}_x), \mathbf{Q}), \tag{2.17a}$$

$$\mathbf{y}_t \mid \mathbf{x}_t \sim p(\mathbf{y}_t \mid \mathbf{x}_t, \boldsymbol{\theta}_y). \tag{2.17b}$$

The Gaussian process prior over $f(\mathbf{x}_t)$, equation (2.16a), can model systematic departures from the nonlinear parametric transition function $\tilde{f}(\mathbf{x}_t, \boldsymbol{\theta}_x)$, equation (2.17a). In practice, one can encode in $m_f(\mathbf{x}_t)$ prior knowledge about the transition dynamics. For example, in the case of modelling a physical system, $m_f(\mathbf{x}_t)$ can be based on equations of the underlying physics. But those equations need not perfectly describe the system. Dynamics not modelled in $m_f(\mathbf{x}_t)$ can be captured in the posterior over $f(\mathbf{x}_t)$.

Following the theorem in Section 3.2.1, this model is equivalent to the models in Sections 2.3.4 and 2.3.5. However, in general, converting the models in Sections 2.3.4 and 2.3.5 to a "transition-only GP-SSM" form requires an increase in the dimensionality of the state-space from $\dim(\mathbf{x}_t)$ to $\dim(\mathbf{x}_t) + \dim(\mathbf{y}_t)$.

Inference and Learning: Frigola et al. (2013) derived a factorised formulation of the (non-Gaussian) prior over state trajectories $p(\mathbf{x}_{0:T})$ in the form of a product of Gaussians. This formulation made possible the use of a Particle Markov Chain Monte Carlo (PMCMC) approach especially suited to non-Markovian models. In this approach, the joint smoothing posterior $p(\mathbf{x}_{0:T}|\mathbf{y}_{0:T})$ is sampled directly without the need to know $f(\mathbf{x})$ which has been marginalised.

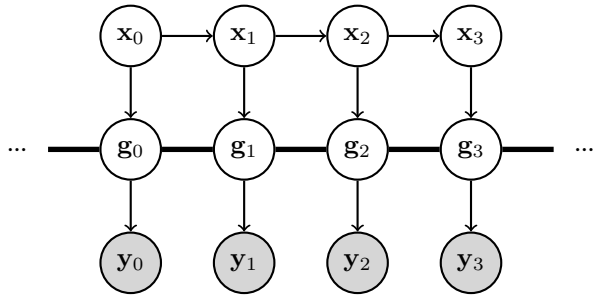
Note that this is markedly different to the conventional approach in parametric models where the smoothing distribution is obtained conditioned on a model of the dynamics. A related approach seeking a maximum (marginal) likelihood estimate of the hyper-parameters via Stochastic Approximation EM was presented in (Frigola et al., 2014b). Finding point estimates of the hyper-parameters can be particularly useful when it is not obvious how to specify a prior over those parameters, e.g. for the inducing input locations in sparse GPs. Chapter 4 provides an expanded exposition of those learning methods based on PMCMC.

McHutchon and Rasmussen (2014) and McHutchon (2014) used a *parametric* model for the state transition function inspired by the form of a GP regression posterior akin to that presented in (Turner et al., 2010). They compared several inference and learning schemes based on analytic approximations and sampling. Learning was performed by finding maximum likelihood estimates of the parameters of the state transition function and the hyper-parameters.

Wu et al. (2014) used a state-space model with transition GP to model volatilities in a financial time-series setting. They presented an online inference and learning algorithm based on particle filtering.

2.3.4 STATE-SPACE MODEL WITH EMISSION GP

Graphical model:



Generative model:

$$g(\mathbf{x}) \sim \mathcal{GP}\big(m_g(\mathbf{x}), k_g(\mathbf{x}, \mathbf{x}')\big), \tag{2.18a}$$

$$\mathbf{x}_0 \sim p(\mathbf{x}_0), \tag{2.18b}$$

$$\mathbf{x}_{t+1} \mid \mathbf{x}_t \sim p(\mathbf{x}_{t+1} \mid \mathbf{x}_t, \boldsymbol{\theta}_x), \tag{2.18c}$$

$$\mathbf{g}_t = g(\mathbf{x}_t), \tag{2.18d}$$

$$\mathbf{y}_t \mid \mathbf{g}_t \sim p(\mathbf{y}_t \mid \mathbf{g}_t, \boldsymbol{\theta}_y). \tag{2.18e}$$

Description: As opposed to the state-space model with transition GP, in this model the state transition has a parametric form whereas a GP prior is placed over the emission function $g(\mathbf{x})$.

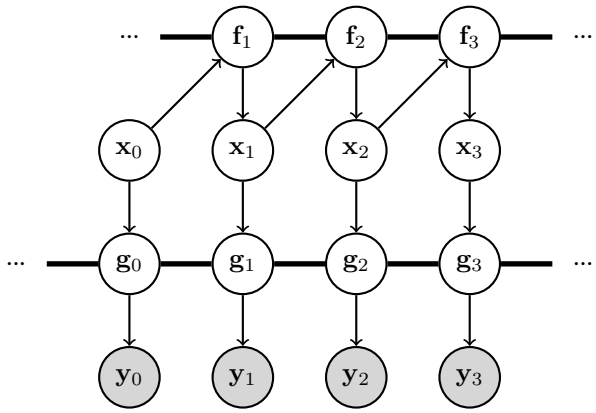
If $p(\mathbf{y}_t \mid \mathbf{g}_t, \boldsymbol{\theta}_y) = \mathcal{N}(\mathbf{y}_t \mid \mathbf{g}_t, \mathbf{R})$ it is possible to analytically marginalise $g(\mathbf{x})$ to obtain a Gaussian likelihood $p(\mathbf{y}_{0:T} \mid \mathbf{x}_{0:T})$ with a potentially non-diagonal covariance matrix.

Following the theorem in Section 3.2.1, this model is equivalent to the transition GP model (Section 2.3.3). Moreover, if the parametric state transition in this model is linear-Gaussian, it can be considered a special case of the GP-LVM with GP on the latent variables (Section 2.3.7).

Inference and Learning: Ferris et al. (2007) introduced this model as a GP-LVM with a Markovian parametric density on the latent variables. The model was learnt by finding maximum a posteriori (MAP) estimates of the states.

2.3.5 STATE-SPACE MODEL WITH TRANSITION AND EMISSION GPS

Graphical model:



Generative model:

$$f(\mathbf{x}) \sim \mathcal{GP}\big(m_f(\mathbf{x}), k_f(\mathbf{x}, \mathbf{x}')\big), \tag{2.19a}$$

$$g(\mathbf{x}) \sim \mathcal{GP}\big(m_g(\mathbf{x}), k_g(\mathbf{x}, \mathbf{x}')\big), \tag{2.19b}$$

$$\mathbf{x}_0 \sim p(\mathbf{x}_0), \tag{2.19c}$$

$$\mathbf{f}_t = f(\mathbf{x}_{t-1}), \tag{2.19d}$$

$$\mathbf{x}_t \mid \mathbf{f}_t \sim \mathcal{N}(\mathbf{f}_t, \mathbf{Q}), \tag{2.19e}$$

$$\mathbf{g}_t = g(\mathbf{x}_t), \tag{2.19f}$$

$$\mathbf{y}_t \mid \mathbf{g}_t \sim \mathcal{N}(\mathbf{g}_t, \mathbf{R}). \tag{2.19g}$$

Description: This model results from a combination of the models in Sections 2.3.3 and 2.3.4. However, perhaps surprisingly, the theorem presented in Section 3.2.1 shows that this model is equivalent to the transition GP model (Section 2.3.3). Therefore, inference methods designed for the transition GP model can be used here after a straightforward redefinition of the state variable.

Placing GP priors over both the state transition and the emission functions opens the door to non-identifiability issues. Those can be mitigated to some extent by a judicious choice of the parametrisation of the mean and covariance functions (or by placing priors over those hyper-parameters). However, if the latent states do not correspond to any interpretable quantity and the only goal is prediction, the non-identifiability is a problem of lesser importance.

Wang et al. (2006, 2008) presented this model in terms of the weight-space view of Gaussian processes (Rasmussen and Williams, 2006) where an infinite number of weights are marginalised. The expression for $p(\mathbf{y}_{0:T}|\mathbf{x}_{0:T})$ is straightforward since it corresponds to the Gaussian distribution describing the marginal likelihood of the GP regression model. However, $p(\mathbf{x}_{0:T})$ is, in their own words, "more subtle" since a regression approach would lead to "the nonsensical expression $p(\mathbf{x}_2, ..., \mathbf{x}_N|\mathbf{x}_1, ..., \mathbf{x}_{N-1})$". After marginalisation,

Wang et al. obtain an analytic expression for the non-Gaussian $p(\mathbf{x}_{0:T})$. In Chapter 3 of this dissertation, we provide an alternative derivation of $p(\mathbf{x}_{0:T})$ which leads to a number of insights on the model that we use to derive new learning algorithms.

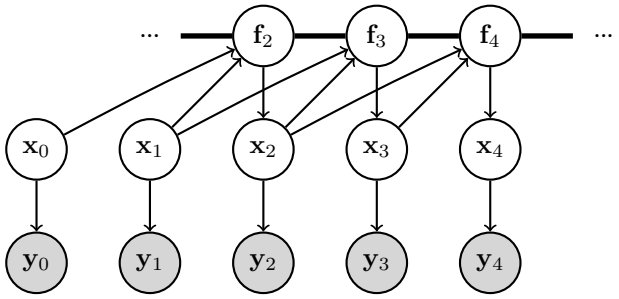
Inference and Learning: Wang et al. (2006) obtained a MAP estimate of the latent state trajectory. Later, in (Wang et al., 2008), they report that MAP learning is prone to overfitting for high-dimensional states and suggest two solutions: 1) a non-probabilistic regularisation of the latent variables, and 2) maximising the marginal likelihood with Monte Carlo Expectation Maximisation where the joint smoothing distribution is sampled using Hamiltonian Monte Carlo.

Ko and Fox (2011) use weak labels of the state to obtain a MAP estimate of the state trajectory and hyper-parameters. The weak labels of the state are actually direct observations of the state contaminated with Gaussian noise. Somehow, this is not the spirit of the model which should "discover" what the state representation is given any sort of observations. In other words, weak labels are nothing else than a particular type of observation that gives a glimpse into what the state actually is. (Note that Ko and Fox (2011) has a crucial technical problem in its Equation 14 which leads to the expression $p(\mathbf{x}_2, ..., \mathbf{x}_{N-1}, \mathbf{x}_N | \mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_{N-1})$ that Wang et al. (2006) had successfully sidestepped. In Chapter 3 of this thesis we provide a novel formulation of the model that sheds light on this issue.)

Turner et al. (2010) presented an approach to learn a model based on the state-space model with transition an emission GPs. They propose a parametric model for the latent functions $f(\mathbf{x})$ and $g(\mathbf{x})$ that takes the form of a posterior from Gaussian process regression whose input and output "datasets" are parameters to be tuned. A point estimate of the parameters for $f(\mathbf{x})$ and $g(\mathbf{x})$ is learnt via maximum likelihood with Expectation Maximisation while treating the state trajectory as latent variables. Several approaches for filtering and smoothing in this kind of models have been developed (Deisenroth et al., 2012; Deisenroth and Mohamed, 2012; McHutchon, 2014). A related model was presented in (Ghahramani and Roweis, 1999) where the nonlinear latent functions take the shape of Gaussian radial basis functions (RBFs) and learning is performed with the EM algorithm using an Extended Kalman Smoother for the E-step.

2.3.6 NON-MARKOVIAN-STATE MODEL WITH TRANSITION GP

Graphical model:



Second order model, i.e. $\mathbf{X}_{t-1} = \{\mathbf{x}_{t-1}, \mathbf{x}_{t-2}\}$.

Generative model:

$$f(\mathbf{X}) \sim \mathcal{GP}\big(m_f(\mathbf{X}), k_f(\mathbf{X}, \mathbf{X}')\big), \tag{2.20a}$$

$$\mathbf{X}_{\tau_x - 1} \sim p(\mathbf{X}_{\tau_x - 1}), \tag{2.20b}$$

$$\mathbf{f}_t = f(\mathbf{X}_{t-1}), \tag{2.20c}$$

$$\mathbf{x}_t \mid \mathbf{f}_t \sim \mathcal{N}(\mathbf{f}_t, \mathbf{Q}), \tag{2.20d}$$

$$\mathbf{y}_t \mid \mathbf{x}_t \sim p(\mathbf{y}_t \mid \mathbf{x}_t, \boldsymbol{\theta}_y), \tag{2.20e}$$

where

$$\mathbf{X}_{t-1} = \{\mathbf{x}_{t-1}, ..., \mathbf{x}_{t-\tau_x}\}. \tag{2.20f}$$

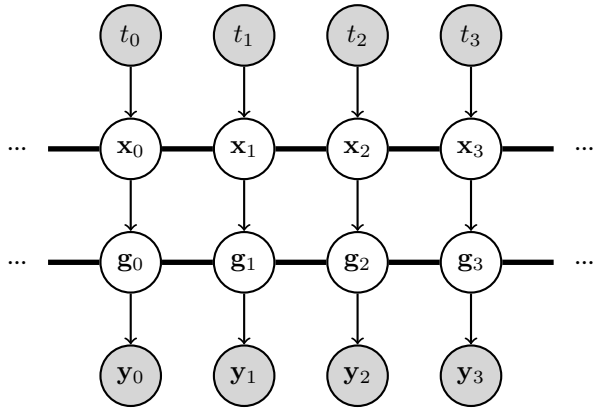
Description: This model is equivalent to a Markovian SSM (Section 2.3.3) with a new state defined as $\mathbf{z}_t = \{\mathbf{x}_t, ..., \mathbf{x}_{t-\tau_x+1}\}$. The reason for explicitly including this model in the present review is that it can also be seen as a generalisation of a nonlinear auto-regressive model (Section 2.3.2) with observation noise.

Inference and Learning: As opposed to auto-regressive models, the present model explicitly takes measurement noise into account. The observations $\mathbf{y}$ can be considered noisy versions of some latent, uncorrupted, variables $\mathbf{x}$. Observation noise in Equation (2.20e) does not affect the future trajectory of the system. Rigorous inference and learning in this model is as computationally demanding as for the equivalent Markovian SSM (Section 2.3.3). However, in (Frigola and Rasmussen, 2013) we proposed a fast approximate method that simultaneously filters $\mathbf{y}$ to approximate $\mathbf{x}$ and learns an auto-regressive model on the approximate $\mathbf{x}$. Any parameter from the preprocessing stage (e.g. the cut-off frequency of a low pass filter) is optimised jointly with the hyper-parameters of the model. See Chapter 6 for more details.

Another approach is to use GP regression with uncertain inputs (McHutchon and Rasmussen, 2011; Damianou and Lawrence, 2015) to learn the auto-regressive function. Note that this is also an approximation since it does not take into account that some inputs to the regression problem are also outputs, e.g. $\mathbf{x}_2$ above is simultaneously a regression output of $f(\mathbf{x}_0, \mathbf{x}_1)$ and an input in $f(\mathbf{x}_1, \mathbf{x}_2)$.

2.3.7 GP-LVM WITH GP ON THE LATENT VARIABLES

Graphical model:



Generative model:

$$x(t) \sim \mathcal{GP}\big(m_x(t), k_x(t,t')\big), \tag{2.21a}$$
$$g(\mathbf{x}) \sim \mathcal{GP}\big(m_g(\mathbf{x}), k_g(\mathbf{x},\mathbf{x}')\big), \tag{2.21b}$$
$$\mathbf{x}_t = x(t), \tag{2.21c}$$
$$\mathbf{g}_t = g(\mathbf{x}_t), \tag{2.21d}$$
$$\mathbf{y}_t \mid \mathbf{g}_t \sim \mathcal{N}\big(\mathbf{g}_t, \beta^{-1}\mathbf{I}\big). \tag{2.21e}$$

Description: This model can be interpreted as a particular case of a Gaussian process latent variable model (GP-LVM, Lawrence (2005)) where the spherical prior over latent variables has been substituted by a second Gaussian process. This Gaussian process provides temporal correlations between the latent variables which become a low-dimensional representation of the high-dimensional observations $\mathbf{y}$.

There is no explicit modelling of a nonlinear transition between successive latent states $\mathbf{x}_t$. However, the use of a kernel $k_x(t,t')$ such as the Ornstein-Uhlenbeck covariance function would result in dynamics on the latent states equivalent to those of a linear-Gaussian state-space model. Unfortunately, nonlinear state transitions, which are one of the strengths of GP-based state-space models, can not be modelled.

Inference and Learning: Lawrence and Moore (2007) introduced this model and sought a maximum a posteriori (MAP) solution with respect to the latent states $\mathbf{x}_t$. More recently, Damianou et al. (2011) proposed a variational learning algorithm that builds on the techniques developed for the Variational Bayesian GP-LVM of Titsias and Lawrence (2010). This approach does not suffer from the overfitting problems of the MAP solution and automatically determines the dimensionality of the latent space when $k_g(\mathbf{x},\mathbf{x}')$ is chosen to be an automatic relevance determination (ARD) kernel. An extension of this work to models with deeper hierarchies has resulted in deep Gaussian processes (Damianou and Lawrence, 2013).

Latent force models (Alvarez et al., 2009, 2013) can also be interpreted as a particular case of GP-LVM with a GP on the latent variables. The variables $\mathbf{x}$ in the top layer are the latent forces whereas the $\mathbf{g}$ layer encodes the linear-Gaussian stochastic differential equation of the latent force model.

2.4 WHY GAUSSIAN PROCESS STATE-SPACE MODELS?

Gaussian Process State-Space Models are particularly appealing because they enjoy the generality of nonlinear state-space models together with the flexible prior over the transition function provided by the Gaussian process. As opposed to auto-regressive models, the presence of a latent state allows for a succinct representation of the dynamics in the form of a Markov chain. The state needs to contain only the information about the system that is essential to determine its future trajectory. As a consequence, discovering a state representation for a system provides useful insights about its nature since it decouples the observations that happen to be available from the dynamics.

A related advantage of state-space models over auto-regressive ones is that observation noise can be explicitly taken into account. To train an auto-regressive model, a time series is broken down into a set of input/output pairs and the function mapping inputs to outputs is learnt with regression techniques. One could use noise-in-the-inputs regression (also known as errors-in-variables) to deal with observation noise. However, this would fail to exploit the fact that the particular noise realisation affecting the observation $\mathbf{y}_t$ is the same when using $\mathbf{y}_t$ as an input or as an output. When learning a state-space model, the time series is not broken down into input/output pairs and inference and learning are performed in a way that coherently takes into account observation noise. This will be clearer in the learning algorithms of the following chapters.

Chapter 3

Gaussian Process State-Space Models – Description

"Trying to understand a hidden Markov model from its observed time series is like trying to figure out the workings of a noisy machine from looking at the shadows its moving parts cast on a wall, with the proviso that the shadows are cast by a randomly-flickering candle."

Shalizi (2008)

This chapter presents a novel formalisation of Gaussian Process State-Space Models (GP-SSMs). In particular, we go beyond describing the joint probability of the model and provide a practical approach to generate samples from its prior. The insights provided by the new description of GP-SSMs will be exploited in the next chapters to derive learning and inference methods adapted to the characteristics of the model. In addition, we present an equivalence result stating that GP-SSMs with GP priors on the transition and emission functions can be reformulated into GP-SSMs with a GP only on the state transition function.

3.1 GP-SSM WITH STATE TRANSITION GP

As presented in Section 2.3, there exist many flavours of GP-SSMs. For clarity, in this section we shall restrict ourselves to a GP-SSM which has a Gaussian process prior on the transition function but a parametric density to describe the likelihood (i.e. the model introduced in Section 2.3.3). This particular model contains the main feature that has made GP-SSMs hard to describe rigorously: the states are simultaneously inputs and outputs of the state transition function. In other words, we want to learn a function whose inputs and outputs are not only latent but also connected in a chain-like manner.

Given an observed time series $\{\mathbf{y}_1, ..., \mathbf{y}_T\}$, we construct a stochastic model

that attempts to explain it by defining a chain of latent states $\{\mathbf{x}_0, ..., \mathbf{x}_T\}$

$$\mathbf{x}_t \mid \mathbf{x}_{t-1} \sim \mathcal{N}(\mathbf{x}_t \mid f(\mathbf{x}_{t-1}), \mathbf{Q}), \tag{3.1a}$$

$$\mathbf{y}_t \mid \mathbf{x}_t \sim p(\mathbf{y}_t \mid \mathbf{x}_t, \boldsymbol{\theta}_y), \tag{3.1b}$$

where $f(\cdot)$ in an unknown nonlinear state transition function and $\mathbf{Q}$ and $\boldsymbol{\theta}_y$ are parameters of the state transition density and likelihood respectively. A common way to learn the unknown state transition function is to define a parametric form for it and proceed to learn its parameters. Common choices for parametric state transition functions are linear functions (Shumway and Stoffer, 1982), radial basis functions (RBFs) (Ghahramani and Roweis, 1999) and other types of neural networks.

GP-SSMs do not restrict the state transition function to a particular parameterised class of functions. Instead, they place a Gaussian process prior over the infinite-dimensional space of functions. This prior can encode assumptions such as continuity or smoothness. In fact, with an appropriate choice of the covariance function, the GP prior puts probability mass over all continuous functions (Micchelli et al., 2006). In other words, as opposed to parameterised functions, GPs do not restrict a priori the class of continuous functions that can be learnt provided that one uses covariance functions that are universal kernels (Micchelli et al., 2006). The squared exponential is an example of such a kernel.

The generative model for a GP-SSM with a GP prior over the state transition function and an arbitrary parametric likelihood is

$$f(\mathbf{x}) \sim \mathcal{GP}\big(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}')\big), \tag{3.2a}$$

$$\mathbf{x}_0 \sim p(\mathbf{x}_0), \tag{3.2b}$$

$$\mathbf{f}_t = f(\mathbf{x}_{t-1}), \tag{3.2c}$$

$$\mathbf{x}_t \mid \mathbf{f}_t \sim \mathcal{N}(\mathbf{x}_t \mid \mathbf{f}_t, \mathbf{Q}), \tag{3.2d}$$

$$\mathbf{y}_t \mid \mathbf{x}_t \sim p(\mathbf{y}_t \mid \mathbf{x}_t, \boldsymbol{\theta}_y). \tag{3.2e}$$

In order to get an intuitive understanding of this model, it can be useful to devise a method to generate synthetic data from it. An idealised approach to generate data from a GP-SSM could consist in first sampling a state transition function (i.e. an infinite-dimensional object) from the GP. This only needs to happen once. Then the full sequence of states can be generated by first sampling the initial state $\mathbf{x}_0$ and then sequentially sampling the rest of the chain $\{\mathbf{f}_1, \mathbf{x}_1, ..., \mathbf{f}_T, \mathbf{x}_T\}$. Each observation $\mathbf{y}_t$ can be sampled conditioned on its corresponding state $\mathbf{x}_t$.

However, this idealised sampling procedure can not be implemented in practice. The reason for this is that it is not possible to sample an infinite-dimensional function and store it in memory. Instead, we will go back to the definition of a Gaussian process and define a practical sampling procedure for the GP-SSM.

A Gaussian process is defined as a collection of random variables, any finite

number of which have a joint Gaussian distribution (Rasmussen and Williams, 2006). However, in a GP-SSM, there is additional structure linking the variables. A sequential sampling procedure provides insight into what this structure is. To sample $\mathbf{f}_t$, instead of conditioning on an infinite-dimensional function, we condition only on the transitions $\{\mathbf{x}_{i-1}, \mathbf{f}_i\}_{i=1}^{t-1}$ seen up to that point. For a zero-mean GP

$$\mathbf{x}_0 \sim p(\mathbf{x}_0), \tag{3.3a}$$
$$\mathbf{f}_1 \mid \mathbf{x}_0 \sim \mathcal{N}\left(\mathbf{f}_1 \mid \mathbf{0}, k(\mathbf{x}_0, \mathbf{x}_0)\right), \tag{3.3b}$$
$$\mathbf{x}_1 \mid \mathbf{f}_1 \sim \mathcal{N}(\mathbf{x}_1 \mid \mathbf{f}_1, \mathbf{Q}), \tag{3.3c}$$
$$\mathbf{f}_2 \mid \mathbf{f}_1, \mathbf{x}_{0:1} \sim \mathcal{N}(\mathbf{f}_2 \mid k(\mathbf{x}_1, \mathbf{x}_0)k(\mathbf{x}_0, \mathbf{x}_0)^{-1}\mathbf{f}_1, \; k(\mathbf{x}_1, \mathbf{x}_1) - k(\mathbf{x}_1, \mathbf{x}_0)k(\mathbf{x}_0, \mathbf{x}_0)^{-1}k(\mathbf{x}_0, \mathbf{x}_1)), \tag{3.3d}$$
$$\mathbf{x}_2 \mid \mathbf{f}_2 \sim \mathcal{N}(\mathbf{x}_2 \mid \mathbf{f}_2, \mathbf{Q}), \tag{3.3e}$$
$$\mathbf{f}_3 \mid \mathbf{f}_{1:2}, \mathbf{x}_{0:2} \sim \mathcal{N}(\mathbf{f}_3 \mid \begin{bmatrix} k(\mathbf{x}_2, \mathbf{x}_0) & k(\mathbf{x}_2, \mathbf{x}_1) \end{bmatrix} \begin{bmatrix} k(\mathbf{x}_0, \mathbf{x}_0) & k(\mathbf{x}_0, \mathbf{x}_1) \\ k(\mathbf{x}_1, \mathbf{x}_0) & k(\mathbf{x}_1, \mathbf{x}_1) \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{f}_1 \\ \mathbf{f}_2 \end{bmatrix}, \; k(\mathbf{x}_2, \mathbf{x}_2) - \begin{bmatrix} k(\mathbf{x}_2, \mathbf{x}_0) & k(\mathbf{x}_2, \mathbf{x}_1) \end{bmatrix} \begin{bmatrix} k(\mathbf{x}_0, \mathbf{x}_0) & k(\mathbf{x}_0, \mathbf{x}_1) \\ k(\mathbf{x}_1, \mathbf{x}_0) & k(\mathbf{x}_1, \mathbf{x}_1) \end{bmatrix}^{-1} \begin{bmatrix} k(\mathbf{x}_0, \mathbf{x}_2) \\ k(\mathbf{x}_1, \mathbf{x}_2) \end{bmatrix}), \tag{3.3f}$$
$$\vdots$$

Since this notation is very cumbersome, we will use the shorthand notation for kernel matrices

$$\mathbf{K}_{i:j,k:l} \triangleq \begin{bmatrix} k(\mathbf{x}_i, \mathbf{x}_k) & \dots & k(\mathbf{x}_i, \mathbf{x}_l) \\ \vdots & & \vdots \\ k(\mathbf{x}_j, \mathbf{x}_k) & \dots & k(\mathbf{x}_j, \mathbf{x}_l) \end{bmatrix}, \tag{3.4}$$

and $\mathbf{K}_{i:j} \triangleq \mathbf{K}_{i:j,i:j}$. Note that the covariance function $k(\cdot, \cdot)$ returns a matrix of the same size as the state in an analogous manner to multi-output Gaussian processes (Rasmussen and Williams, 2006). With this notation, equations (3.3) become

$$\mathbf{x}_0 \sim p(\mathbf{x}_0), \tag{3.5a}$$
$$\mathbf{f}_1 \mid \mathbf{x}_0 \sim \mathcal{N}\left(\mathbf{f}_1 \mid \mathbf{0}, \mathbf{K}_0\right), \tag{3.5b}$$
$$\mathbf{x}_1 \mid \mathbf{f}_1 \sim \mathcal{N}(\mathbf{x}_1 \mid \mathbf{f}_1, \mathbf{Q}), \tag{3.5c}$$
$$\mathbf{f}_2 \mid \mathbf{f}_1, \mathbf{x}_{0:1} \sim \mathcal{N}(\mathbf{f}_2 \mid \mathbf{K}_{1,0}\mathbf{K}_0^{-1}\mathbf{f}_1, \; \mathbf{K}_{1,1} - \mathbf{K}_{1,0}\mathbf{K}_0^{-1}\mathbf{K}_{0,1}), \tag{3.5d}$$
$$\mathbf{x}_2 \mid \mathbf{f}_2 \sim \mathcal{N}(\mathbf{x}_2 \mid \mathbf{f}_2, \mathbf{Q}), \tag{3.5e}$$
$$\mathbf{f}_3 \mid \mathbf{f}_{1:2}, \mathbf{x}_{0:2} \sim \mathcal{N}(\mathbf{f}_3 \mid \mathbf{K}_{2,0:1}\mathbf{K}_{0:1}^{-1}\mathbf{f}_{1:2}, \; \mathbf{K}_{2,2} - \mathbf{K}_{2,0:1}\mathbf{K}_{0:1}^{-1}\mathbf{K}_{0:1,2}), \tag{3.5f}$$
$$\vdots$$

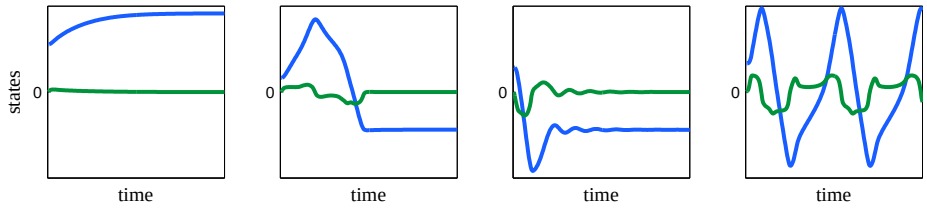


Figure 3.1: State trajectories from four 2-state nonlinear dynamical systems sampled from a GP-SSM prior with identical hyperparameters. The same prior generates systems with qualitatively different behaviours, e.g. the leftmost panel shows behaviour similar to that of a non-oscillatory linear system whereas the rightmost panel appears to have arisen from a limit cycle in a nonlinear system.

For a general time step, the key density is

$$p(\mathbf{f}_t \mid \mathbf{f}_{1:t-1}, \mathbf{x}_{0:t-1}) = \mathcal{N}(\mathbf{f}_t \mid \mathbf{K}_{t-1,0:t-2}\mathbf{K}_{0:t-2}^{-1}\mathbf{f}_{1:t-1}, \\ \mathbf{K}_{t-1,t-1} - \mathbf{K}_{t-1,0:t-2}\mathbf{K}_{0:t-2}^{-1}\mathbf{K}_{0:t-2,t-1}), \quad (3.6)$$

which is analogous to the predictive density in GP regression. In particular, it corresponds to the prediction at a single test point $\mathbf{x}_{t-1}$ using inputs $\mathbf{x}_{0:t-2}$ and *noiseless* outputs $\mathbf{f}_{1:t-1}$.

The joint distribution over latent and observed variables can be expressed as

$$p(\mathbf{y}_{1:T}, \mathbf{x}_{0:T}, \mathbf{f}_{1:T}) = p(\mathbf{x}_0) \prod_{t=1}^{T} p(\mathbf{y}_t \mid \mathbf{x}_t)\, p(\mathbf{x}_t \mid \mathbf{f}_t)\, p(\mathbf{f}_t \mid \mathbf{f}_{1:t-1}, \mathbf{x}_{0:t-1}). \quad (3.7)$$

The behaviour of GP-SSMs is demonstrated in Figure 3.1 which shows four independent state trajectories sampled from a GP-SSM. Equations (3.5) provided the sequential procedure used for sampling. The richness of the nonlinear dynamics allowed by a squared exponential kernel allow for very different qualitative behaviour.

3.1.1 AN IMPORTANT REMARK

At this point, it is useful to note that

$$\prod_{t=1}^{T} p(\mathbf{f}_t \mid \mathbf{f}_{1:t-1}, \mathbf{x}_{0:t-1}) = \mathcal{N}\big(\mathbf{f}_{1:T} \mid \mathbf{0}, \mathbf{K}_{0:T-1}\big). \quad (3.8)$$

Equation (3.8) provides a "telescopic" expression where each $\mathbf{f}_t$ depends on the previous $\mathbf{f}_{1:t-1}$ and $\mathbf{x}_{0:t-1}$. The conditioning on $\mathbf{x}$ only up to time $t-1$ is critical. If we were to condition on the full state trajectory $\mathbf{x}_{0:T}$, the distribution over $\mathbf{f}_{1:T}$ would be markedly different. In particular, we would be facing a problem akin to GP regression where we would be looking at the posterior distribution over the latent function values conditioned on inputs $\mathbf{x}_{0:T-1}$ and *noisy* outputs

$\mathbf{x}_{1:T}$

$$p(\mathbf{f}_{1:T} \mid \mathbf{x}_{0:T}) = \mathcal{N}\big(\mathbf{f}_{1:T} \mid \mathbf{K}_{0:T-1}\widetilde{\mathbf{K}}_{0:T-1}^{-1}\mathbf{x}_{1:T}, \\ \mathbf{K}_{0:T-1} - \mathbf{K}_{0:T-1}\widetilde{\mathbf{K}}_{0:T-1}^{-1}\mathbf{K}_{0:T-1}^{\top}\big), \tag{3.9}$$

with $\widetilde{\mathbf{K}}_{0:T-1} \triangleq \mathbf{K}_{0:T-1} + \mathbf{I}_T \otimes \mathbf{Q}$. As one could expect, this expression reduces to $\mathbf{f}_{1:T} = \mathbf{x}_{1:T}$ if $\mathbf{Q} = \mathbf{0}$.

We can summarise the previous argument by stating that

$$\prod_{t=1}^{T} p(\mathbf{f}_t \mid \mathbf{f}_{1:t-1}, \mathbf{x}_{0:t-1}) = \mathcal{N}\big(\mathbf{f}_{1:T} \mid \mathbf{0}, \mathbf{K}_{0:T-1}\big) \neq p(\mathbf{f}_{1:T} \mid \mathbf{x}_{0:T}). \tag{3.10}$$

For completeness, we provide the expressions for an arbitrary mean function where we use $\mathbf{m}_{0:t} \triangleq [m(\mathbf{x}_0)^\top, ..., m(\mathbf{x}_t)^\top]^\top$:

$$\prod_{t=1}^{T} p(\mathbf{f}_t \mid \mathbf{f}_{1:t-1}, \mathbf{x}_{0:t-1}) = \mathcal{N}\big(\mathbf{f}_{1:T} \mid \mathbf{m}_{0:T-1}, \mathbf{K}_{0:T-1}\big), \tag{3.11}$$

and

$$p(\mathbf{f}_{1:T} \mid \mathbf{x}_{0:T}) = \mathcal{N}\big(\mathbf{f}_{1:T} \mid \mathbf{m}_{0:T-1} + \mathbf{K}_{0:T-1}\widetilde{\mathbf{K}}_{0:T-1}^{-1}(\mathbf{x}_{1:T} - \mathbf{m}_{0:T-1}), \\ \mathbf{K}_{0:T-1} - \mathbf{K}_{0:T-1}\widetilde{\mathbf{K}}_{0:T-1}^{-1}\mathbf{K}_{0:T-1}^{\top}\big). \tag{3.12}$$

3.1.2 MARGINALISATION OF $\mathbf{f}_{1:T}$

By marginalising out the variables $\mathbf{f}_{1:T}$ we will obtain a prior distribution over state trajectories that will be particularly useful for learning GP-SSMs with sample-based approaches (Chapter 4). The density over latent variables in a GP-SSM is

$$p(\mathbf{x}_{0:T}, \mathbf{f}_{1:T}) = p(\mathbf{x}_0) \prod_{t=1}^{T} p(\mathbf{x}_t \mid \mathbf{f}_t)\, p(\mathbf{f}_t \mid \mathbf{f}_{1:t-1}, \mathbf{x}_{0:t-1}) \tag{3.13a}$$

$$= p(\mathbf{x}_0)\, \mathcal{N}(\mathbf{x}_{1:T} \mid \mathbf{f}_{1:T}, \mathbf{I}_T \otimes \mathbf{Q})\, \mathcal{N}(\mathbf{f}_{1:T} \mid \mathbf{m}_{0:T-1}, \mathbf{K}_{0:T-1}). \tag{3.13b}$$

Marginalising with respect to $\mathbf{f}_{1:T}$ we obtain the prior distribution over the latent state trajectory

$$p(\mathbf{x}_{0:T}) = \int p(\mathbf{x}_{0:T}, \mathbf{f}_{1:T})\, \mathrm{d}\mathbf{f}_{1:T} \tag{3.14a}$$

$$= p(\mathbf{x}_0) \int \mathcal{N}(\mathbf{x}_{1:T} \mid \mathbf{f}_{1:T}, \mathbf{I}_T \otimes \mathbf{Q})\, \mathcal{N}(\mathbf{f}_{1:T} \mid \mathbf{m}_{0:T-1}, \mathbf{K}_{0:T-1})\, \mathrm{d}\mathbf{f}_{1:T} \tag{3.14b}$$

$$= p(\mathbf{x}_0)|(2\pi)^{n_x T}\widetilde{\mathbf{K}}_{0:T-1}|^{-\frac{1}{2}} \exp(-\frac{1}{2}(\mathbf{x}_{1:T} - \mathbf{m}_{0:T-1})^\top \widetilde{\mathbf{K}}_{0:T-1}^{-1}(\mathbf{x}_{1:T} - \mathbf{m}_{0:T-1})), \tag{3.14c}$$

which is the density provided in (Wang et al., 2006) albeit in a slightly different form.

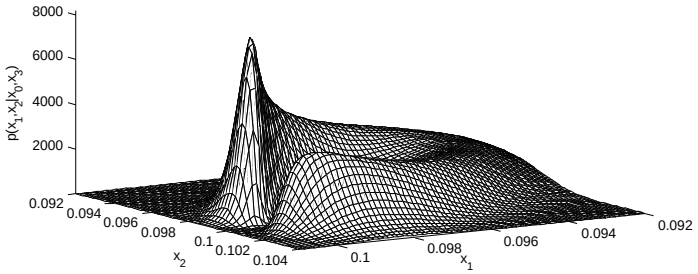


Figure 3.2: Density function $p(x_{1:2} \mid x_0 = 0.1, x_3 = 0.095)$ for a one-state GP-SSM.

To solve the integral in equation 3.14b we have used a standard result from integrating a product of Gaussians. However, the resulting distribution is *not* Gaussian. The fact that $\mathbf{K}_{0:T-1}$ already depends on $\mathbf{x}_{0:T-1}$ results in equation (3.14c) not being Gaussian in $\mathbf{x}_{0:T}$. This is clear in Figure 3.2 which shows how the conditional distributions of $p(x_{0:3})$ are far from Gaussian.

Although the prior distribution over state trajectories $p(\mathbf{x}_{0:T})$ is not Gaussian, it can be expressed as a product of Gaussians

$$p(\mathbf{x}_{0:T}) = p(\mathbf{x}_0) \prod_{t=1}^{T} p(\mathbf{x}_t \mid \mathbf{x}_{0:t-1}) \tag{3.15a}$$

$$= p(\mathbf{x}_0) \prod_{t=1}^{T} \mathcal{N}\big(\mathbf{x}_t \mid \boldsymbol{\mu}_t(\mathbf{x}_{0:t-1}), \boldsymbol{\Sigma}_t(\mathbf{x}_{0:t-1})\big), \tag{3.15b}$$

with

$$\boldsymbol{\mu}_t(\mathbf{x}_{0:t-1}) = \mathbf{m}_{t-1} + \mathbf{K}_{t-1,0:t-2}\widetilde{\mathbf{K}}_{0:t-2}^{-1}\,(\mathbf{x}_{1:t-1} - \mathbf{m}_{0:t-2}), \tag{3.15c}$$

$$\boldsymbol{\Sigma}_t(\mathbf{x}_{0:t-1}) = \widetilde{\mathbf{K}}_{t-1} - \mathbf{K}_{t-1,0:t-2}\widetilde{\mathbf{K}}_{0:t-2}^{-1}\mathbf{K}_{t-1,0:t-2}^{\top} \tag{3.15d}$$

for $t \geq 2$ and $\boldsymbol{\mu}_1(\mathbf{x}_0) = \mathbf{m}_0$, $\boldsymbol{\Sigma}_1(\mathbf{x}_0) = \widetilde{\mathbf{K}}_0$. Equation (3.15b) follows from the fact that, once conditioned on $\mathbf{x}_{0:t-1}$, the distribution over $\mathbf{x}_t$ corresponds to the predictive density of GP regression at a single test point $\mathbf{x}_{t-1}$ conditioned on inputs $\mathbf{x}_{0:t-2}$ and *noisy* outputs $\mathbf{x}_{1:t-1}$. Note that naive evaluation of Equation (3.15) has complexity $\mathcal{O}(T^4)$ whereas Equation (3.14) can be evaluated straightforwardly in $\mathcal{O}(T^3)$.

3.1.3 MARGINALISATION OF $f(\mathbf{x})$

So far in this chapter, we have provided a constructive method to obtain the joint probability density of a GP-SSM. This approach sidestepped having to deal with infinite-dimensional objects. Here, following (Turner et al., 2015), we re-derive the density over the latent variables of a GP-SSM by starting from a joint distribution containing a Gaussian process and marginalising it to obtain a joint distribution over a finite number of variables. The goal of the presentation below is to provide an intuition for the marginalisation of the latent function $f(\cdot)$. Strictly speaking, however, conventional integration over an infinite dimensional object such as $f(\cdot)$ is not possible and the derivation below should

be considered just a sketch. We refer the reader to (Matthews et al., 2015) for a much more technical exposition of integration of infinite dimensional objects in the context of sparse Gaussian processes.

We start with a joint distribution of the state trajectory and the random function $f(\cdot)$

$$p(\mathbf{x}_{0:T}, f(\cdot)) = p(f(\cdot))\, p(\mathbf{x}_0) \prod_{t=1}^{T} p(\mathbf{x}_t|\mathbf{x}_{t-1}, f(\cdot), \mathbf{Q}), \tag{3.16}$$

and introduce new variables $\mathbf{f}_t = f(\mathbf{x}_{t-1})$ which, in probabilistic terms can be defined as a Dirac delta $p(\mathbf{f}_t \mid f(\cdot), \mathbf{x}_{t-1}) = \delta(\mathbf{f}_t - f(\mathbf{x}_{t-1}))$.

$$p(\mathbf{x}_{0:T}, \mathbf{f}_{1:T}, f(\cdot)) = p(f(\cdot))\, p(\mathbf{x}_0) \prod_{t=1}^{T} p(\mathbf{x}_t|\mathbf{f}_t, \mathbf{Q}) p(\mathbf{f}_t \mid f(\cdot), \mathbf{x}_{t-1}), \tag{3.17}$$

By integrating the latent function at every point other than $\mathbf{x}_{0:T-1}$ we obtain

$$\begin{aligned} p(\mathbf{x}_{0:T}, \mathbf{f}_{1:T}) &= \int p(\mathbf{x}_{0:T}, \mathbf{f}_{1:T}, f(\cdot))\, \mathrm{d}f_{\setminus \mathbf{x}_{0:T-1}} && \text{(3.18a)} \\ &= p(\mathbf{x}_0) \left(\prod_{t=1}^{T} \mathcal{N}(\mathbf{x}_t|\mathbf{f}_t, \mathbf{Q}) \right) \int p(f(\cdot)) \prod_{t=1}^{T} \delta(\mathbf{f}_t - f(\mathbf{x}_{t-1}))\, \mathrm{d}f_{\setminus \mathbf{x}_{0:T-1}} && \text{(3.18b)} \\ &= p(\mathbf{x}_0)\, \mathcal{N}(\mathbf{x}_{1:T} \mid \mathbf{f}_{1:T}, \mathbf{I}_T \otimes \mathbf{Q})\, \mathcal{N}(\mathbf{f}_{1:T} \mid \mathbf{m}_{0:T-1}, \mathbf{K}_{0:T-1}), && \text{(3.18c)} \end{aligned}$$

which recovers the joint density from equation (3.13b)

3.2 GP-SSM WITH TRANSITION AND EMISSION GPS

GP-SSMs containing a nonlinear emission function with a GP prior have also been studied in the literature (see Section 2.3.5). In such a model, the nonlinear function $g(\mathbf{x}_t)$ in the state-space model

$$\begin{aligned} \mathbf{x}_t \mid \mathbf{x}_{t-1} &\sim \mathcal{N}(\mathbf{x}_t \mid f(\mathbf{x}_{t-1}), \mathbf{Q}), && \text{(3.19a)} \\ \mathbf{y}_t \mid \mathbf{x}_t &\sim \mathcal{N}(\mathbf{y}_t \mid g(\mathbf{x}_t), \mathbf{R}), && \text{(3.19b)} \end{aligned}$$

also has a GP prior which results in a generative model described by

$$\begin{aligned} f(\mathbf{x}) &\sim \mathcal{GP}\big(m_f(\mathbf{x}), k_f(\mathbf{x}, \mathbf{x}')\big), && \text{(3.20a)} \\ g(\mathbf{x}) &\sim \mathcal{GP}\big(m_g(\mathbf{x}), k_g(\mathbf{x}, \mathbf{x}')\big), && \text{(3.20b)} \\ \mathbf{x}_0 &\sim p(\mathbf{x}_0), && \text{(3.20c)} \\ \mathbf{f}_t &= f(\mathbf{x}_{t-1}), && \text{(3.20d)} \\ \mathbf{x}_t \mid \mathbf{f}_t &\sim \mathcal{N}(\mathbf{x}_t \mid \mathbf{f}_t, \mathbf{Q}), && \text{(3.20e)} \\ \mathbf{g}_t &= g(\mathbf{x}_t), && \text{(3.20f)} \\ \mathbf{y}_t \mid \mathbf{g}_t &\sim \mathcal{N}(\mathbf{y}_t \mid \mathbf{g}_t, \mathbf{R}). && \text{(3.20g)} \end{aligned}$$

This variant of GP-SSM is a straightforward extension to the one with a parametric observation model. The distribution over latent states and transition function values is the same. However, an observation at time t is *not* conditionally independent of the rest of the observations given $\mathbf{x}_t$. The joint probability distribution of this model is

$$p(\mathbf{y}_{1:T}, \mathbf{g}_{1:T}, \mathbf{x}_{0:T}, \mathbf{f}_{1:T}) = p(\mathbf{x}_{0:T}, \mathbf{f}_{1:T})\, p(\mathbf{g}_{1:T}|\mathbf{x}_{1:T}) \prod_{t=1}^{T} p(\mathbf{y}_t|\mathbf{g}_t) \tag{3.21a}$$

$$= p(\mathbf{x}_{0:T}, \mathbf{f}_{1:T})\, \mathcal{N}(\mathbf{g}_{1:T} \mid \mathbf{m}^{(g)}_{1:T}, \mathbf{K}^{(g)}_{1:T}) \prod_{t=1}^{T} \mathcal{N}(\mathbf{y}_t \mid \mathbf{g}_t, \mathbf{R}). \tag{3.21b}$$

Where $p(\mathbf{x}_{0:T}, \mathbf{f}_{1:T})$ is the same as in Equation (3.7) and $\mathbf{m}^{(g)}$ and $\mathbf{K}^{(g)}$ denote the mean function vector and kernel matrix using the mean and covariance functions in Equation (3.20b).

3.2.1 EQUIVALENCE BETWEEN GP-SSMS

In the following, we present an equivalence result between the two GP-SSM variants introduced in this chapter. We show how the model with nonlinearities for both the transition and emission functions (Sec. 2.3.5 and 3.2) is equivalent to a model with a redefined state-space but which only has a nonlinear function in the state transition (Sec. 2.3.3 and 3.1).

We consider state-space models with independent and identically distributed additive noise

$$\mathbf{x}_{t+1} = f(\mathbf{x}_t) + \mathbf{v}_t, \tag{3.22a}$$

$$\mathbf{y}_t = g(\mathbf{x}_t) + \mathbf{e}_t. \tag{3.22b}$$

In GP-SSMs, Gaussian process priors are placed over f and g in order to perform inference on those functions.

THEOREM: The model in (3.22) is a particular case of the following model

$$\mathbf{z}_{t+1} = h(\mathbf{z}_t) + \mathbf{w}_t, \tag{3.23a}$$

$$\mathbf{y}_t = \boldsymbol{\gamma}_t + \mathbf{e}_t, \tag{3.23b}$$

where $\mathbf{z}_t \triangleq (\boldsymbol{\xi}_t, \boldsymbol{\gamma}_t)$ is the *augmented state*.

PROOF: Consider the case where

$$\mathbf{z}_{t+1} = (\boldsymbol{\xi}_{t+1}, \boldsymbol{\gamma}_{t+1}) = \big(f(\boldsymbol{\xi}_t), g(\boldsymbol{\xi}_t)\big) + (\boldsymbol{\nu}_t, 0). \tag{3.24}$$

Then, the model in (3.23) can be written as

$$\boldsymbol{\xi}_{t+1} = f(\boldsymbol{\xi}_t) + \boldsymbol{\nu}_t, \tag{3.25a}$$
$$\boldsymbol{\gamma}_{t+1} = g(\boldsymbol{\xi}_t), \tag{3.25b}$$
$$\mathbf{y}_t = \boldsymbol{\gamma}_t + \mathbf{e}_t. \tag{3.25c}$$

If we now take $\boldsymbol{\xi}_t = \mathbf{x}_{t+1}$ and $\boldsymbol{\nu}_t = \mathbf{v}_{t+1}$ and substitute them in (3.25) we obtain

$$\mathbf{x}_{t+1} = f(\mathbf{x}_t) + \mathbf{v}_t, \tag{3.26a}$$
$$\mathbf{y}_t = g(\mathbf{x}_t) + \mathbf{e}_t. \tag{3.26b}$$

COROLLARY: A learning procedure that is able to learn an unknown state transition function can also be used to simultaneously learn *both* $f(\mathbf{x}_t)$ and $g(\mathbf{x}_t)$.

3.3 SPARSE GP-SSMS

Gaussian processes are very useful priors over functions that, in comparison with parametric models, place very few assumptions on the shapes of the unknown functions. However, this richness comes at a price: computational requirements for training and prediction scale unfavourably with the size of the training set. To alleviate this problem, methods relying on *sparse Gaussian processes* have been developed that retain many of the advantages of vanilla Gaussian processes while reducing the computational cost (Quiñonero-Candela and Rasmussen, 2005; Titsias, 2009; Hensman et al., 2013, 2015).

Sparse Gaussian processes often rely on a number of inducing points. Those inducing *points*, denoted by $\mathbf{u}_i$, are values of the unknown function at arbitrary locations named inducing *inputs*

$$\mathbf{u}_i = f(\mathbf{z}_i). \tag{3.27}$$

For conciseness of notation, we define the set with all inducing points $\mathbf{u} \triangleq \{\mathbf{u}_i\}_{i=1}^M$ and the set of their respective inducing inputs $\mathbf{z} \triangleq \{\mathbf{z}_i\}_{i=1}^M$. If we place a Gaussian process prior over a function f, the inducing points are jointly Gaussian with values of the latent function at any other location

$$p(\mathbf{f}, \mathbf{u}) = \mathcal{N}\left(\begin{bmatrix}\mathbf{f}\\ \mathbf{u}\end{bmatrix} \mid \begin{bmatrix}\mathbf{m}_\mathbf{x}\\ \mathbf{m}_\mathbf{z}\end{bmatrix}, \begin{bmatrix}\mathbf{K}_\mathbf{x} & \mathbf{K}_{\mathbf{x},\mathbf{z}}\\ \mathbf{K}_{\mathbf{z},\mathbf{x}} & \mathbf{K}_\mathbf{z}\end{bmatrix}\right). \tag{3.28}$$

This method of explicitly representing the values of the function at an additional finite number of input locations $\mathbf{z}$ is sometimes called augmentation. However, one could argue that the model has not been augmented since the inducing points were "already there"; we are simply assigning a variable name to the value of the function at input locations $\mathbf{z}$.

For regression, a variety of sparse GP methods have been developed by

making additional assumptions on the relationships between function values at the training, test and inducing inputs (Quiñonero-Candela and Rasmussen, 2005). More recently, a variational approach to sparse GPs (Titsias, 2009) has resulted in sparse GP methods for a number of models (Titsias and Lawrence, 2010; Damianou et al., 2011; Hensman et al., 2013, 2015).

In Chapter 5 we will develop a variational inference approach for GP-SSMs. It is then useful at this point to provide an explicit description of a GP-SSM with inducing points. The latent variables of such a GP-SSM have the following joint density

$$p(\mathbf{x}_{0:T}, \mathbf{f}_{1:T}, \mathbf{u} \mid \mathbf{z}) = p(\mathbf{u} \mid \mathbf{z})\, p(\mathbf{x}_0) \prod_{t=1}^{T} p(\mathbf{x}_t \mid \mathbf{f}_t)\, p(\mathbf{f}_t | \mathbf{f}_{1:t-1}, \mathbf{x}_{0:t-1}, \mathbf{u}, \mathbf{z}), \quad (3.29)$$

where $p(\mathbf{u}|\mathbf{z}) = \mathcal{N}(\mathbf{m}_\mathbf{z}, \mathbf{K}_\mathbf{z})$. To lighten the notation, from now on we will omit the explicit conditioning on inducing inputs $\mathbf{z}$. The product of latent function conditionals can be succinctly represented by the following Gaussian

$$\prod_{t=1}^{T} p(\mathbf{f}_t | \mathbf{f}_{1:t-1}, \mathbf{x}_{0:t-1}, \mathbf{u}) = \mathcal{N}\big(\mathbf{f}_{1:T} \,|\mathbf{m}_{0:T-1} + \mathbf{K}_{0:T-1,\mathbf{z}}\mathbf{K}_\mathbf{z}^{-1}(\mathbf{u} - \mathbf{m}_\mathbf{z}), \\ \mathbf{K}_{0:T-1} - \mathbf{K}_{0:T-1,\mathbf{z}}\mathbf{K}_\mathbf{z}^{-1}\mathbf{K}_{0:T-1,\mathbf{z}}^\top\big), \quad (3.30)$$

which is equivalent to the predictive density of GP regression with inputs $\mathbf{z}$, *noiseless* outputs $\mathbf{u}$ and test inputs $\mathbf{x}_{0:T-1}$. We emphasise again that this telescopic product of conditionals is *not* the same as

$$p(\mathbf{f}_{1:T}|\mathbf{x}_{0:T-1}, \mathbf{u}) = \mathcal{N}\big(\mathbf{m}_{0:T-1} + \\ \mathbf{K}_{0:T-1,\{\mathbf{z},0:T-2\}}(\mathbf{K}_{\{\mathbf{z},0:T-2\}} + \mathbf{\Sigma}_\mathbf{Q})^{-1} \begin{pmatrix} \mathbf{u} - \mathbf{m}_\mathbf{z} \\ \mathbf{x}_{1:T-1} - \mathbf{m}_{0:T-2} \end{pmatrix}, \\ \mathbf{K}_{0:T-1} - \mathbf{K}_{0:T-1,\{\mathbf{z},0:T-2\}}(\mathbf{K}_{\{\mathbf{z},0:T-2\}} + \mathbf{\Sigma}_\mathbf{Q})^{-1}\mathbf{K}_{0:T-1,\{\mathbf{z},0:T-2\}}^\top\big) \quad (3.31)$$

where $\mathbf{\Sigma}_\mathbf{Q} = \text{blockdiag}(\mathbf{0}, \mathbf{I} \otimes \mathbf{Q})$ takes into account that $\mathbf{u}$ are equivalent to noiseless observations whereas neighbouring states are affected by process noise with covariance $\mathbf{Q}$.

In Section 5.7.1 we will highlight a number of parallelisms between sparse GP-SSMs and Recurrent Neural Networks: a class of neural networks that is currently receiving attention due to its ability to learn insightful representations of sequences (Sutskever, 2013; LeCun et al., 2015).

3.4 SUMMARY OF GP-SSM DENSITIES

Table 3.1 summarises the key densities for GP-SSMs introduced so far.

$p(\mathbf{y}_{1:T}, \mathbf{x}_{0:T}, \mathbf{f}_{1:T})$	Eq. (3.7)
$\prod_{t=1}^{T} p(\mathbf{f}_t \vert \mathbf{f}_{1:t-1}, \mathbf{x}_{0:t-1})$	Eq. (3.11)
$p(\mathbf{f}_{1:T} \vert \mathbf{x}_{0:T})$	Eq. (3.12)
$p(\mathbf{x}_{0:T}, \mathbf{f}_{1:T})$	Eq. (3.13b)
$p(\mathbf{x}_{0:T}, f(\cdot))$	Eq. (3.16)
$p(\mathbf{x}_{0:T})$	Eqs. (3.14) & (3.15)
$p(\mathbf{x}_{0:T}, \mathbf{f}_{1:T}, \mathbf{u} \mid \mathbf{z})$	Eqs. (3.29)
$\prod_{t=1}^{T} p(\mathbf{f}_t \vert \mathbf{f}_{1:t-1}, \mathbf{x}_{0:t-1}, \mathbf{u})$	Eq. (3.30)
$p(\mathbf{f}_{1:T} \vert \mathbf{x}_{0:T-1}, \mathbf{u})$	Eq. (3.31)

Table 3.1: Summary of GP-SSM densities.

Chapter 4

Gaussian Process State-Space Models – Monte Carlo Learning

This chapter exploits insights from the novel description of Gaussian Process State-Space Models presented so far to derive Bayesian learning methods based on Monte Carlo sampling. In particular, we present a fully Bayesian approach to jointly sample from all variables and a related Empirical Bayes method that finds maximum likelihood estimates of the hyper-parameters while approximately marginalising the latent state trajectories.

This chapter introduces new learning methods for GP-SSMs originally published in (Frigola et al., 2013) and (Frigola et al., 2014b) that rely on the new expression for the prior density over the state trajectory expressed as a product of Gaussians (Equation 3.15b).

4.1 Introduction

The sampling methods presented in this chapter make heavy use of recent advances in Particle Markov Chain Monte Carlo (PMCMC). In contrast to prior work on learning GP-SSMs, we do not restrict ourselves to the case where the dimensionality of the state-space is much lower than that of the observation space.

The key feature of our approach is that we sample from the smoothing distribution *while the nonlinear dynamics are marginalised out*. In other words, samples from the posterior over state trajectories are obtained without having to specify the dynamics of the system. This contrasts with conventional approaches to smoothing where the smoothing distribution is conditioned on fixed dynamics (Barber et al., 2011; Murphy, 2012).

The main advantage of Monte Carlo methods over approaches such as variational inference (Chapter 5) is that they do not rely on assumptions about the

shape of the posterior. Therefore, they are useful as a gold standard against which to compare other learning algorithms. In terms of practical application, however, they tend to suffer from longer computation time, in particular when doing predictions.

4.2 FULLY BAYESIAN LEARNING

This section introduces a fully Bayesian approach to learning GP-SSMs. Learning the state transition function in a GP-SSM is particularly challenging because the states are not observed. The goal is to learn a function where both its inputs and outputs are latent variables. Most previous approaches, see (McHutchon, 2014) for an overview, hinge on learning a parametric approximation to a GP-SSM. We address this problem in a fundamentally different way that keeps alive all the nonparametric richness of the model.

Our approach is based on the key observation that learning the transition function in a GP-SSM is equivalent to finding the *smoothing distribution* in that model. In other words, once the smoothing distribution $p(\mathbf{x}_{0:T} \mid \mathbf{y}_{1:T}, \boldsymbol{\theta})$ has been found, the posterior over the state transition function can be straightforwardly computed. The predictive distribution over $\mathbf{f}_* = f(\mathbf{x}_*)$, evaluated at an arbitrary test point $\mathbf{x}_*$, is

$$p(\mathbf{f}_* \mid \mathbf{x}_*, \mathbf{y}_{1:T}, \boldsymbol{\theta}) = \int p(\mathbf{f}_* \mid \mathbf{x}_*, \mathbf{x}_{0:T}, \boldsymbol{\theta})\, p(\mathbf{x}_{0:T} \mid \mathbf{y}_{1:T}, \boldsymbol{\theta})\, \mathrm{d}\mathbf{x}_{0:T}, \tag{4.1}$$

where the distribution $p(\mathbf{f}_* \mid \mathbf{x}_*, \mathbf{x}_{0:T}, \boldsymbol{\theta})$ is equivalent to the predictive distribution in Gaussian process regression using $\mathbf{x}_{0:T-1}$ as inputs and $\mathbf{x}_{1:T}$ as *noisy* outputs.

In practice, there are a number of parameters in a GP-SSM that we would like to learn from data: hyper-parameters of the GP, covariance matrix $\mathbf{Q}$, etc. We employ a blocked Gibbs sampling approach which alternates between sampling from the smoothing distribution $p(\mathbf{x}_{0:T}|\mathbf{y}_{1:T}, \boldsymbol{\theta})$ and from the hyper-parameters $p(\boldsymbol{\theta}|\mathbf{y}_{1:T}, \mathbf{x}_{0:T})$. This results in a joint posterior distribution over state trajectories and parameters $p(\mathbf{x}_{0:T}, \boldsymbol{\theta}|\mathbf{y}_{1:T})$ that can be used to make predictions (Section 4.2.3).

4.2.1 SAMPLING STATE TRAJECTORIES WITH PMCMC

To obtain samples from $p(\mathbf{x}_{0:T} \mid \mathbf{y}_{1:T}, \boldsymbol{\theta})$ we propose the use of Particle Markov Chain Monte Carlo (PMCMC, Andrieu et al. (2010) and Appendix A.1). PMCMC is an influential recent technique that is particularly suited to sampling from high-dimensional and highly correlated distributions.

One of the challenges of applying Monte Carlo methods to high dimensional distributions with complex patterns of dependence is the design of appropriate proposal distributions. The essence of PMCMC is the use of Sequential Monte Carlo to generate proposals of the highly correlated variables, in our case, the state trajectory. Samples obtained with PMCMC can be interpreted as

standard MCMC samples which leave the target density invariant for any fixed number of particles greater than one. PMCMC algorithms are exact approximations to idealised MCMC algorithms targeting the smoothing distribution (Andrieu et al., 2010).

PMCMC has a number of appealing properties:

- As opposed to Sequential Monte Carlo (SMC) methods, PMCMC provides true samples from the posterior distribution. SMC relies on importance sampling of unnormalised distributions and would require an infinite number of particles to be unbiased (Doucet and Johansen, 2011).

- PMCMC suffers less from path degeneracy than SMC. Path degeneracy is the consequence of successive resampling in SMC that results in low particle diversity at time points far from the final time instant (Lindsten and Schön, 2013).

- PMCMC is simpler to tune by non-experts than Hamiltonian Monte-Carlo. As a consequence, it is more amenable to automated learning. Also, it does not require derivatives with respect to the latent variables (i.e. the high-dimensional state trajectory).

To compute the smoothing distribution of a GP-SSM, we employ a specific variant of PMCMC known as Particle Gibbs with Ancestor Sampling (PGAS, Lindsten et al. (2014)) which we describe in more detail in Appendix A.1.1. PGAS is particularly suited to non-Markovian models such as a GP-SSM where

$$p(\mathbf{x}_{0:T}) = p(\mathbf{x}_0) \prod_{t=1}^{T} \mathcal{N}\big(\mathbf{x}_t \mid \boldsymbol{\mu}_t(\mathbf{x}_{0:t-1}), \boldsymbol{\Sigma}_t(\mathbf{x}_{0:t-1})\big).$$

The PGAS algorithm has other appealing qualities such as good performance with a small number of particles, the ability to avoid a smoothing backward pass on the time series and a simpler implementation than the Particle Gibbs with Backwards Simulation algorithm (Lindsten and Schön, 2013; McHutchon, 2014).

The PGAS algorithm requires us to be able to:

1. Sample from $p(\mathbf{x}_0 \mid \boldsymbol{\theta})$.

2. Sample from $p(\mathbf{x}_t \mid \mathbf{x}_{0:t-1}, \boldsymbol{\theta})$, Equation (3.15).

3. Evaluate the density $p(\mathbf{x}_{t:T} \mid \mathbf{x}_{0:t-1}, \boldsymbol{\theta}) = \frac{p(\mathbf{x}_{0:T}|\boldsymbol{\theta})}{p(\mathbf{x}_{0:t-1}|\boldsymbol{\theta})}$, Equation (3.14).

4. Evaluate the density $p(\mathbf{y}_t \mid \mathbf{x}_t, \boldsymbol{\theta})$.

All these operations are straightforward when using the GP-SSM formulation introduced in Chapter 3. Leaving the technical details for Appendix A.1, Algorithm 1 shows the necessary steps to learn GP-SSMs with PGAS.

Algorithm 1 Particle Gibbs with Ancestor Sampling (PGAS) for GP-SSMs

1. Set $\boldsymbol{\theta}[0]$ and $\mathbf{x}_{0:T}[0]$ arbitrarily.
2. **for** $\ell = 1$ **to** L **do**
   (a) Draw $\boldsymbol{\theta}[\ell]$ conditionally on $\mathbf{x}_{0:T}[\ell-1]$ and $\mathbf{y}_{1:T}$ (Section 4.2.2).
   (b) Run a Conditional Particle Filter with Ancestor Sampling (Appendix A.1.1) targeting $p(\mathbf{x}_{0:T} \mid \boldsymbol{\theta}[\ell], \mathbf{y}_{1:T})$, conditionally on $\mathbf{x}_{0:T}[\ell-1]$:
      i. Set $\widetilde{\mathbf{x}}_{0:T} = \mathbf{x}_{0:T}[\ell-1]$, the reference trajectory.
      ii. Draw $\mathbf{x}_0^i \sim p(\mathbf{x}_0 \mid \boldsymbol{\theta}[\ell])$ for $i = 1, ..., N-1$.
      iii. Set $\mathbf{x}_0^N = \widetilde{\mathbf{x}}_0$.
      iv. Set $\mathbf{w}_0^i = 1/N$ for $i = 1, ..., N-1$.
      v. **for** $t = 1$ **to** T **do**
         A. Draw $\mathbf{a}_t^i$ with $\mathrm{P}(\mathbf{a}_t^i = j) = \mathbf{w}_{t-1}^j$ for $i = 1, ..., N-1$.
         B. Draw $\mathbf{x}_t^i \sim p(\mathbf{x}_t \mid \mathbf{x}_{0:t-1}^{\mathbf{a}_t^i}, \boldsymbol{\theta}[\ell])$ for $i = 1, ..., N-1$. (Eq. 3.15b)
         C. Draw $\mathbf{a}_t^N$ with $\mathrm{P}(\mathbf{a}_t^N = j) \propto \mathbf{w}_{t-1}^j \; p(\widetilde{\mathbf{x}}_{t:T} \mid \mathbf{x}_{0:t-1}^j, \boldsymbol{\theta}[\ell])$. (Eq. 3.14, see Section 4.4 for an efficient implementation)
         D. Set $\mathbf{x}_t^N = \widetilde{\mathbf{x}}_t$.
         E. Set $\mathbf{w}_t^i = p(\mathbf{y}_t \mid \mathbf{x}_t^i, \boldsymbol{\theta}[\ell])$ for $i = 1, ..., N$
         F. Normalise the weights $\mathbf{w}_t$ to sum to 1.
   (c) Sample k with $\mathrm{P}(k = i) = w_T^i$ and set $\mathbf{x}_{0:T}[\ell] = \mathbf{x}_{0:T}^k$.

4.2.2 SAMPLING THE HYPER-PARAMETERS

The next step in our blocked Gibbs sampling approach is to sample the hyper-parameters given a state trajectory and sequence of observations, i.e. sample from $p(\boldsymbol{\theta} \mid \mathbf{x}_{0:T}, \mathbf{y}_{1:T})$. In the common situation where there are distinct hyper-parameters for the likelihood $p(\mathbf{y}_{1:T} \mid \mathbf{x}_{0:T}, \boldsymbol{\theta}_{\mathbf{y}})$ and for the prior over trajectories $p(\mathbf{x}_{0:T} \mid \boldsymbol{\theta}_{\mathbf{x}})$, and if the prior over the hyper-parameters factorises between those two groups we obtain

$$p(\boldsymbol{\theta} \mid \mathbf{x}_{0:T}, \mathbf{y}_{1:T}) \propto p(\boldsymbol{\theta}_{\mathbf{y}} \mid \mathbf{x}_{0:T}, \mathbf{y}_{1:T}) \; p(\boldsymbol{\theta}_{\mathbf{x}} \mid \mathbf{x}_{0:T}). \tag{4.2}$$

We can thus proceed to sample the two groups of hyper-parameters independently. Sampling $\boldsymbol{\theta}_{\mathbf{y}}$ is straightforward in most cases, especially if conjugate priors for the likelihood are used. Sampling $\boldsymbol{\theta}_{\mathbf{x}}$ will, nevertheless, be harder since the covariance function hyper-parameters enter the expression in a non-trivial way. However, we note that once the state trajectory is fixed, we are left with a problem analogous to Gaussian process regression where $\mathbf{x}_{0:T-1}$ are the inputs, $\mathbf{x}_{1:T}$ are the *noisy* outputs and $\mathbf{Q}$ is the likelihood covariance matrix. Given that the latent dynamics can be marginalised out analytically, sampling the hyper-parameters with slice sampling (Neal, 2003) is straightforward for GP-based models (Agarwal and Gelfand, 2005).

4.2.3 MAKING PREDICTIONS

The predictive distribution for the value of the transition function at a particular test point $\mathbf{x}_*$ is

$$p(\mathbf{f}_* \mid \mathbf{x}_*, \mathbf{y}_{1:T}) = \int p(\mathbf{f}_* \mid \mathbf{x}_*, \mathbf{x}_{0:T}, \boldsymbol{\theta})\, p(\mathbf{x}_{0:T}, \boldsymbol{\theta} \mid \mathbf{y}_{1:T})\, \mathrm{d}\mathbf{x}_{0:T}\, \mathrm{d}\boldsymbol{\theta}. \tag{4.3}$$

Using a sample-based approximation of $p(\mathbf{x}_{0:T}, \boldsymbol{\theta} \mid \mathbf{y}_{1:T})$, the predictive distribution of a GP-SSM, Equation (4.14), can be approximated by

$$p(\mathbf{f}_* \mid \mathbf{x}_*, \mathbf{y}_{1:T}) \approx \frac{1}{L} \sum_{\ell=1}^{L} p(\mathbf{f}_* \mid \mathbf{x}_*, \mathbf{x}_{0:T}[\ell], \boldsymbol{\theta}[\ell]) \tag{4.4a}$$

$$= \frac{1}{L} \sum_{\ell=1}^{L} \mathcal{N}(\mathbf{f}_* \mid \boldsymbol{\mu}^{\ell}(\mathbf{x}_*), \boldsymbol{\Sigma}^{\ell}(\mathbf{x}_*)), \tag{4.4b}$$

where L is the number of samples and $\boldsymbol{\mu}^{\ell}(\mathbf{x}_*)$ and $\boldsymbol{\Sigma}^{\ell}(\mathbf{x}_*)$ follow the expressions for the predictive distribution in standard GP regression if $\mathbf{x}_{0:T-1}[\ell]$ are treated as inputs, $\mathbf{x}_{1:T}[\ell]$ are treated as outputs and $\mathbf{Q}[\ell]$ is the likelihood covariance matrix. This mixture of Gaussians is an expressive representation of the predictive density which can, for instance, correctly take into account multimodality arising from ambiguity in the measurements. Although factorised covariance matrices can be pre-computed, the overall computational cost will increase linearly with L and T. This is a consequence of having to average over L samples of trajectories each having length T. The computational cost can be reduced by thinning the Markov chain to use samples that are largely uncorrelated.

A variational inference approach will be presented in Chapter 5 which avoids averaging over samples at prediction time. As a result, it provides faster predictions.

4.2.4 EXPERIMENTS

4.2.4.1 LEARNING A NONLINEAR SYSTEM BENCHMARK

In the following, we use a nonlinear dynamical system benchmark that is popular within the sequential Monte Carlo community (Gordon et al., 1993)

$$x_{t+1} = a x_t + b \frac{x_t}{1 + x_t^2} + c u_t + v_t, \qquad v_t \sim \mathcal{N}(0, q), \tag{4.5a}$$

$$y_t = d x_t^2 + e_t, \qquad e_t \sim \mathcal{N}(0, r), \tag{4.5b}$$

with parameters $(a, b, c, d, q, r) = (0.5, 25, 8, 0.05, 10, 1)$ and a known input $u_t = \cos(1.2(t+1))$. One of the challenging properties of this system is that the quadratic measurement function (4.5b) tends to induce a bimodal distribution in the marginal smoothing distribution. For instance, if we were to consider only one measurement in isolation and $r = 0$ we would have $x_t = \pm\sqrt{\frac{y_t}{d}}$. Moreover, the state transition function (4.5a) exhibits a very sharp gradient in

RMSE	prediction of $\mathbf{f}^*\|\mathbf{x}_t^*, \mathbf{u}_t^*, \text{data}$	smoothing $\mathbf{x}_{0:T}\|\text{data}$
Ground truth model (known parameters)	–	2.7 ± 0.5
GP-SSM (proposed, model B mean function)	1.7 ± 0.2	3.2 ± 0.5
Sparse GP-SSM (proposed, model B mean function)	1.8 ± 0.2	2.7 ± 0.4
Model B (fixed parameters)	7.1 ± 0.0	13.6 ± 1.1
Ground truth model, learnt parameters	0.5 ± 0.2	3.0 ± 0.4
Linear model, learnt parameters	5.5 ± 0.1	6.0 ± 0.5

Table 4.1: RMSE to ground truth values over 10 independent runs for the 1-dimensional nonlinear system benchmark.

the x_t direction at the origin, but is otherwise well behaved as $x_t \to \pm\infty$.

The system is simulated for $T = 200$ time steps, using log-normal priors for the hyper-parameters, and the PGAS sampler is then run for 50 iterations using $N = 20$ particles. To illustrate the capability of the GP-SSM to make use of a parametric model as baseline, we use a mean function with the same parametric form as the true system, but parameters $(a, b, c) = (0.3, 7.5, 0)$. This function, denoted *model B*, is manifestly different to the actual state transition (green vs. black surfaces in Figure 4.2).

Figure 4.1 shows the samples from the smoothing distribution (red). It is apparent that the distribution covers two alternative state trajectories at particular times (e.g. $t = 10$). In Figure 4.2 we plot samples from the smoothing distribution, where each circle corresponds to $(\mathbf{x}_t, \mathbf{u}_t, \mathbb{E}[\mathbf{f}_t])$. Although the parametric model used in the mean function of the GP (green) is clearly not representative of the true dynamics (black), the samples from the smoothing distribution accurately portray the underlying system. The smoothness prior embodied by the GP allows for accurate sampling from the smoothing distribution even when the parametric model of the dynamics fails to capture important features.

To measure the predictive capability of the learnt transition dynamics, we generate a new dataset consisting of $10\,000$ time steps and present the RMSE between the mean of the predicted value of $f(\mathbf{x}_t, \mathbf{u}_t)$ and the actual one. We compare the results from GP-SSM with the predictions obtained from two parametric models (one with the true model structure and one linear model) and two known models (the ground truth model and model B). We also report results for the sparse GP-SSM using an FIC prior with 40 inducing points. Table 4.1 summarises the results, averaged over 10 independent training and test datasets. We also report the RMSE from samples of the joint smoothing distribution to the ground truth trajectory.

4.2.4.2 LEARNING A CART AND POLE SYSTEM

We apply our approach to learn a model of a cart and pole system used in reinforcement learning. The system consists of a cart, with a free-spinning pendulum, rolling on a horizontal track. An external force is applied to the cart. The system's dynamics can be described by four states and a set of nonlinear ordinary differential equations (Deisenroth, 2010). We learn a GP-SSM based on 100 observations of the state corrupted with Gaussian noise. Although the

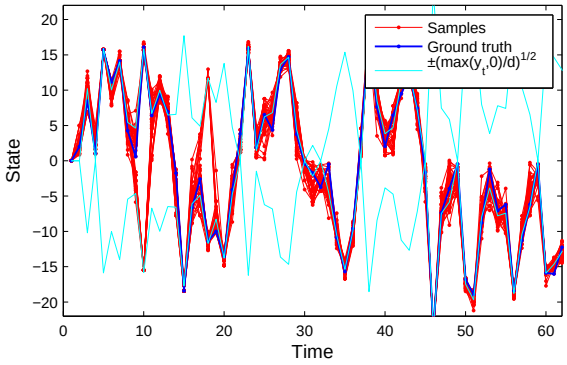


Figure 4.1: Samples from the smoothing distribution. The cyan line shows states that could have given rise to the current observation in the absence of observation noise. Note how in some occasions the posterior distribution is clearly multimodal (e.g. $t = 18$).

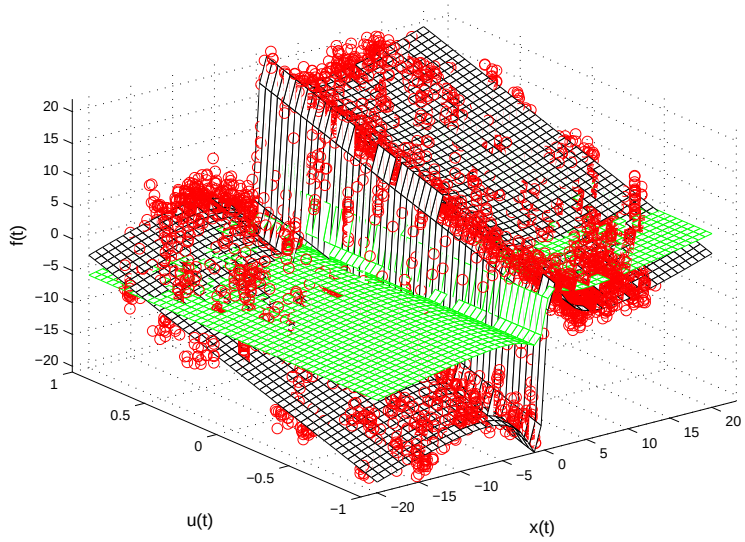


Figure 4.2: State transition function (black: actual transition function, green: mean function (model B) and red: smoothing samples).

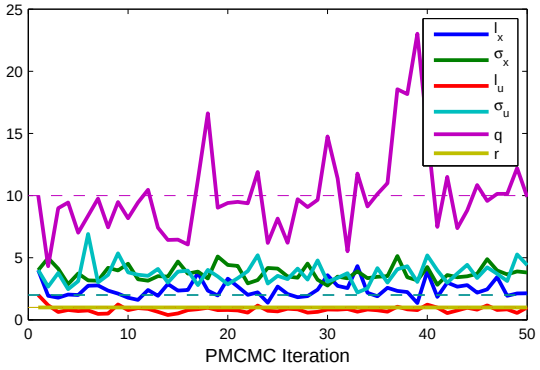


Figure 4.3: Hyper-parameter samples.

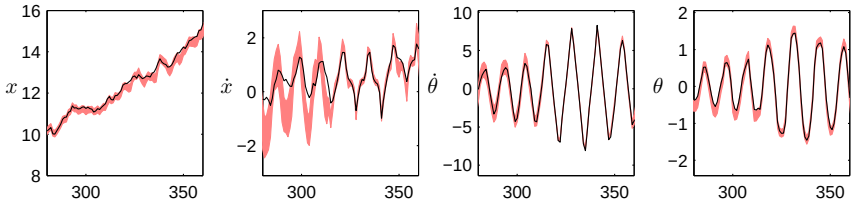


Figure 4.4: One time step ahead predictive distribution for each of the states of the cart and pole system. Black: ground truth. Colored band: one standard deviation from the mixture of Gaussians predictive.

training set only explores a small region of the 4-dimensional state space, we can learn a model of the dynamics which can produce one step ahead predictions such the ones in Figure 4.4. We obtain a predictive distribution in the form of a mixture of Gaussians from which we display the first and second moments. Crucially, the learnt model reports different amounts of uncertainty in different regions of the state-space. For instance, note the narrower error-bars on some states between $t = 320$ and $t = 350$. This is due to the model being more confident in its predictions in areas that are closer to the training data.

4.3 EMPIRICAL BAYES

An alternative approach to the fully Bayesian learning presented in Section 4.2 is to learn the hyper-parameters of the GP-SSM via *maximum likelihood* while approximately marginalising the state trajectory. Maximum likelihood (ML) is a widely used frequentist estimator of the parameters of statistical models. The ML estimator $\widehat{\boldsymbol{\theta}}_{\mathrm{ML}}$ is defined as the value of the parameters that makes the available observations $\mathbf{y}_{1:T}$ as likely as possible

$$\widehat{\boldsymbol{\theta}}_{\mathrm{ML}} = \arg\max_{\boldsymbol{\theta}} \; p(\mathbf{y}_{1:T} \mid \boldsymbol{\theta}). \tag{4.6}$$

The GP-SSM has two types of latent variables that need to be marginalised (integrated out) in order to compute the likelihood

$$\begin{aligned} p(\mathbf{y}_{1:T} \mid \boldsymbol{\theta}) &= \int p(\mathbf{y}_{1:T}, \mathbf{x}_{0:T}, \mathbf{f}_{1:T} \mid \boldsymbol{\theta}) \, \mathrm{d}\mathbf{x}_{0:T} \, \mathrm{d}\mathbf{f}_{1:T} \\ &= \int p(\mathbf{y}_{1:T} \mid \mathbf{x}_{0:T}, \boldsymbol{\theta}) \left(\int p(\mathbf{x}_{0:T}, \mathbf{f}_{1:T} \mid \boldsymbol{\theta}) \, \mathrm{d}\mathbf{f}_{1:T} \right) \mathrm{d}\mathbf{x}_{0:T}. \end{aligned} \tag{4.7}$$

The latent variables $\mathbf{f}_{1:T}$ can be marginalised analytically, Equations (3.14,3.15). This is equivalent to integrating out the uncertainty in the unknown function $f(\mathbf{x})$ and working directly with a prior over the state trajectories $p(\mathbf{x}_{0:T} \mid \boldsymbol{\theta})$. This prior encodes assumptions (e.g. smoothness) about $f(\mathbf{x})$ specified in the Gaussian process prior over functions. Once the latent function has been

marginalised, the likelihood becomes

$$p(\mathbf{y}_{1:T} \mid \boldsymbol{\theta}) = \int p(\mathbf{y}_{1:T} \mid \mathbf{x}_{0:T}, \boldsymbol{\theta})\, p(\mathbf{x}_{0:T} \mid \boldsymbol{\theta})\, \mathrm{d}\mathbf{x}_{0:T}. \tag{4.8}$$

The integration with respect to $\mathbf{x}_{0:T}$, however, is not analytically tractable. This difficulty will be addressed in the following section.

Direct application of maximum likelihood on $p(\mathbf{y}_{1:T}|\mathbf{x}_{0:T}, \boldsymbol{\theta}_{\mathbf{y}})$ to obtain estimates of the state trajectory and likelihood parameters would result in overfitting due to the large dimensionality of the state trajectory. However, by introducing a prior on the state trajectories and marginalising them as in (4.8), we obtain the so-called marginal likelihood. Maximisation of the marginal likelihood with respect to the parameters results in a procedure known as type II maximum likelihood or *empirical Bayes* (Bishop, 2006). Empirical Bayes reduces the risk of over-fitting since it automatically incorporates a trade-off between model fit and model complexity, a property often known as Bayesian Occam's razor (MacKay, 2003; Ghahramani, 2012).

In the context of GP-SSMs, an Empirical Bayes approach can be useful if one has no meaningful prior over some of the hyper-parameters. For instance, this could be the case for pseudo-inputs parameterising a FIC covariance function (Section 4.4.1). If meaningful priors are available, we recommend using the fully Bayesian approach from Section 4.2 since it has the same prediction cost but handles uncertainty in the hyper-parameters.

4.3.1 PARTICLE STOCHASTIC APPROXIMATION EM

As pointed out above, direct evaluation of the marginal likelihood (4.8) is not possible for a GP-SSM. However, by viewing the latent states $\mathbf{x}_{0:T}$ as missing data, we are able to evaluate the *complete data* log-likelihood

$$\log p(\mathbf{y}_{1:T}, \mathbf{x}_{0:T} \mid \boldsymbol{\theta}) = \log p(\mathbf{y}_{1:T} \mid \mathbf{x}_{0:T}, \boldsymbol{\theta}) + \log p(\mathbf{x}_{0:T} \mid \boldsymbol{\theta}). \tag{4.9}$$

We therefore turn to the *Expectation-Maximisation* (EM) algorithm (Dempster et al., 1977). The EM algorithm uses Equation (4.9) to construct a surrogate cost function for the maximum likelihood problem, defined as

$$Q(\boldsymbol{\theta}, \boldsymbol{\theta}') = \int p(\mathbf{x}_{0:T} \mid \mathbf{y}_{1:T}, \boldsymbol{\theta}') \log p(\mathbf{y}_{1:T}, \mathbf{x}_{0:T} \mid \boldsymbol{\theta})\, \mathrm{d}\mathbf{x}_{0:T}. \tag{4.10}$$

Expectation-maximisation is an iterative procedure that maximises (4.8) by iterating two steps, expectation (E) and maximisation (M),

E: Find $Q(\boldsymbol{\theta}, \boldsymbol{\theta}_{k-1})$.

M: Find $\boldsymbol{\theta}_k = \arg\max_{\boldsymbol{\theta}} Q(\boldsymbol{\theta}, \boldsymbol{\theta}_{k-1})$.

The resulting sequence $\{\boldsymbol{\theta}_k\}_{k\geq 0}$ will, under weak assumptions, converge to a local maximum of the marginal likelihood $p(\mathbf{y}_{1:T} \mid \boldsymbol{\theta})$.

To implement the above procedure it is necessary to compute the integral in (4.10), which, in general, is not tractable for a GP-SSM. To deal with this difficulty, we employ a Monte-Carlo-based implementation of the EM algorithm, referred to as *Particle Stochastic Approximation EM* (PSAEM) (Lindsten, 2013). This procedure is a combination of stochastic approximation EM (SAEM) (Delyon et al., 1999) and Particle MCMC (Andrieu et al., 2010). PSAEM is a competitive alternative to particle-smoothing-based EM algorithms as it enjoys better convergence properties and has a much lower computational cost (Lindsten, 2013). The method maintains a stochastic approximation of the auxiliary quantity (4.10), $\widehat{Q}_k(\boldsymbol{\theta}) \approx Q(\boldsymbol{\theta}, \boldsymbol{\theta}_{k-1})$. This approximation is updated according to

$$\widehat{Q}_k(\boldsymbol{\theta}) = (1-\gamma_k)\widehat{Q}_{k-1}(\boldsymbol{\theta}) + \gamma_k \log p(\mathbf{y}_{1:T}, \mathbf{x}_{0:T}[k] \mid \boldsymbol{\theta}). \tag{4.11}$$

Here, $\{\gamma_k\}_{k\geq 0}$ is a sequence of step sizes, satisfying the usual stochastic approximation conditions: $\sum_k \gamma_k = \infty$ and $\sum_k \gamma_k^2 < \infty$ (Robbins and Monro, 1951). A typical choice is to take $\gamma_k = k^{-p}$ with $p \in]0.5, 1]$, where a smaller value of p gives a more rapid convergence at the cost of higher variance. In the vanilla SAEM algorithm, $\mathbf{x}_{0:T}[k]$ is a draw from the smoothing distribution $p(\mathbf{x}_{0:T} \mid \mathbf{y}_{1:T}, \boldsymbol{\theta}_{k-1})$. In this setting, (Delyon et al., 1999) show that using the stochastic approximation (4.11) instead of (4.10) in the EM algorithm results in a valid method, i.e. $\{\boldsymbol{\theta}_k\}_{k\geq 0}$ will still converge to a maximiser of the marginal likelihood $p(\mathbf{y}_{0:T} \mid \boldsymbol{\theta})$.

The PSAEM algorithm is an extension of SAEM which is useful when it is not possible to easily sample from the joint smoothing distribution. This is indeed the case in our setting. Instead of sampling from the smoothing distribution, the sample state trajectory $\mathbf{x}_{0:T}[k]$ in Equation (4.11) may be drawn from an ergodic Markov kernel, leaving the smoothing distribution invariant. Under suitable conditions on the kernel, this will not violate the validity of SAEM, see (Andrieu et al., 2005; Andrieu and Vihola, 2011).

In PSAEM, the Markov kernel on the space of trajectories, denoted as $P^N_{\boldsymbol{\theta}}(\mathbf{x}^\star_{0:T} \mid \widetilde{\mathbf{x}}_{0:T})$, is constructed using PMCMC theory. In particular, we use the Particle Gibbs with Ancestor Sampling algorithm (PGAS, Lindsten et al. (2014)) that we described in Section 4.2. PGAS is a sequential Monte Carlo method, akin to a standard particle filter, but with the difference that one particle at each time point is specified *a priori*. This reference state trajectory, denoted as $\widetilde{\mathbf{x}}_{0:T}$, can be thought of as guiding the particles of the particle filter to the "correct" regions of the state-space. More formally PGAS defines a Markov kernel which leaves the joint smoothing distribution invariant, i.e. for any $\boldsymbol{\theta}$

$$\int P^N_{\boldsymbol{\theta}}(\mathbf{x}^\star_{0:T} \mid \widetilde{\mathbf{x}}_{0:T}) p(\widetilde{\mathbf{x}}_{0:T} \mid \mathbf{y}_{1:T}, \boldsymbol{\theta}) \, \mathrm{d}\widetilde{\mathbf{x}}_{0:T} = p(\mathbf{x}^\star_{0:T} \mid \mathbf{y}_{1:T}, \boldsymbol{\theta}). \tag{4.12}$$

The PGAS kernel is indexed by N, which is the number of particles used in the underlying particle filter. Note that the desired property (4.12) holds for any number of particles, i.e. the number of particles only affects the mixing

Algorithm 2 PSAEM for GP-SSMs

1. Set $\boldsymbol{\theta}_0$ and $\mathbf{x}_{0:T}[0]$ arbitrarily. Set $\widehat{Q}_0(\boldsymbol{\theta}) \equiv 0$.
2. **for** $k \geq 1$ **do**
   (a) Simulate $\mathbf{x}_{0:T}[k] \sim P^N_{\boldsymbol{\theta}_{k-1}}(\cdot \mid \mathbf{x}_{0:T}[k-1])$ (run CPF-AS in Appendix A.1.1 and set $\mathbf{x}_{0:T}[k]$ to one of the particle trajectories sampled according to their importance weights).
   (b) Update $\widehat{Q}_k(\boldsymbol{\theta})$ according to (4.11).
   (c) Compute $\boldsymbol{\theta}_k = \arg\max_{\boldsymbol{\theta}} \widehat{Q}_k(\boldsymbol{\theta})$.

of the Markov kernel. A larger N implies faster mixing, which in turn results in better approximations of the auxiliary quantity (4.11). However, it has been experienced in practice that the correlation between consecutive trajectories drops of quickly as N increases (Lindsten and Schön, 2013; McHutchon, 2014) and, for many models, a moderate N (e.g. in the range 5–20) is enough to get a rapidly mixing kernel.

Next, we address the M-step of the EM algorithm. Maximising the quantity (4.11) will typically not be possible in closed form. Instead, a numerical optimisation routine can be used (e.g. a quasi-Newton method such as BFGS). Using Equation (4.9), the gradient of the complete data log-likelihood can be written as

$$\frac{\partial}{\partial \boldsymbol{\theta}} \log p(\mathbf{y}_{0:T}, \mathbf{x}_{0:T} \mid \boldsymbol{\theta}) = \sum_{t=1}^{T} \frac{\partial}{\partial \boldsymbol{\theta}} \log p(\mathbf{y}_t \mid \mathbf{x}_t, \boldsymbol{\theta}) + \sum_{t=1}^{T} \frac{\partial}{\partial \boldsymbol{\theta}} \log p(\mathbf{x}_t \mid \mathbf{x}_{0:t-1}, \boldsymbol{\theta}) + \frac{\partial}{\partial \boldsymbol{\theta}} \log p(\mathbf{x}_0 \mid \boldsymbol{\theta}), \tag{4.13}$$

The resulting PSAEM algorithm for learning of GP-SSMs is summarised in Algorithm 2.

4.3.2 MAKING PREDICTIONS

Making predictions with a maximum likelihood estimate of the parameters is analogous to the fully Bayesian case except that the parameters are not marginalised out. The predictive distribution for the value of the transition function at a particular test point $\mathbf{x}_*$ is

$$p(\mathbf{f}_* \mid \mathbf{x}_*, \mathbf{y}_{1:T}) \approx \int p(\mathbf{f}_* \mid \mathbf{x}_*, \mathbf{x}_{0:T}, \widehat{\boldsymbol{\theta}}_{\mathrm{ML}})\, p(\mathbf{x}_{0:T} \mid \mathbf{y}_{1:T}, \widehat{\boldsymbol{\theta}}_{\mathrm{ML}})\, \mathrm{d}\mathbf{x}_{0:T}. \tag{4.14}$$

Since we have a sample-based approximation of the posterior over state trajectories $p(\mathbf{x}_{0:T} \mid \mathbf{y}_{1:T})$, the predictive distribution of the GP-SSM can be

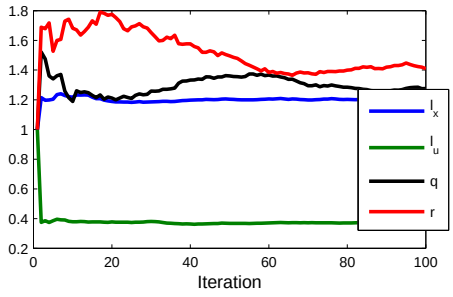


Figure 4.5: Convergence of parameters when learning a linear system using a linear covariance function.

approximated by a mixture of Gaussians

$$p(\mathbf{f}_* \mid \mathbf{x}_*, \mathbf{y}_{1:T}) \approx \frac{1}{L}\sum_{\ell=1}^{L} p(\mathbf{f}_* \mid \mathbf{x}_*, \mathbf{x}_{0:T}[\ell], \widehat{\boldsymbol{\theta}}_{\mathrm{ML}}) \tag{4.15a}$$

$$= \frac{1}{L}\sum_{\ell=1}^{L} \mathcal{N}(\mathbf{f}_* \mid \boldsymbol{\mu}^{\ell}(\mathbf{x}_*), \boldsymbol{\Sigma}^{\ell}(\mathbf{x}_*)). \tag{4.15b}$$

4.3.3 EXPERIMENTS

In this section we present the results of applying PSAEM to learn various dynamical systems.

4.3.3.1 LEARNING A LINEAR SYSTEM

Although GP-SSMs are particularly suited to nonlinear system identification, we start by illustrating the Empirical Bayes learning approach with a simple linear system

$$\mathbf{x}_{t+1} = 0.8\,\mathbf{x}_t + 3\,\mathbf{u}_t + \mathbf{v}_t, \qquad \mathbf{v}_t \sim \mathcal{N}(0, 1.5), \tag{4.16a}$$

$$\mathbf{y}_t = 2\,\mathbf{x}_t + \mathbf{e}_t, \qquad \mathbf{e}_t \sim \mathcal{N}(0, 1.5), \tag{4.16b}$$

excited by a periodic input. The GP-SSM can model this linear system by using a linear covariance function for the GP. This covariance function imposes, in a somewhat indirect fashion, that the state-transition function in Equation (4.16a) must be linear. A GP-SSM with linear covariance function is formally equivalent to a linear state-space model where a Gaussian prior is placed over the unknown parameters (Rasmussen and Williams, 2006, Section 2.1). The hyper-parameters of the covariance function are equivalent to the variances of a zero-mean prior over A and B. Therefore, the application of PSAEM to this particular GP-SSM can be interpreted as finding the hyper-parameters of a Gaussian prior over the parameters of the linear model that maximise the likelihood of the observed data whilst marginalising over A and B. In addition, the likelihood will be simultaneously optimised with respect to the process noise and measurement noise variances (q and r respectively).

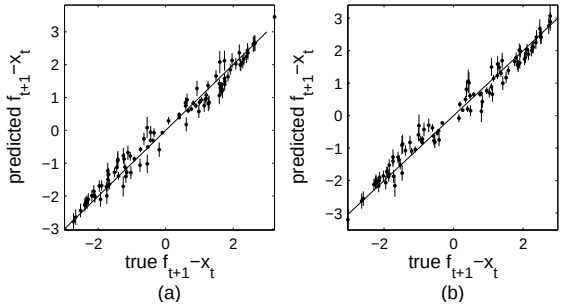


Figure 4.6: Linear dynamical system learnt using a GP-SSM with linear covariance function. Predictions (a) on training data, and (b) on test data (see text for more details).

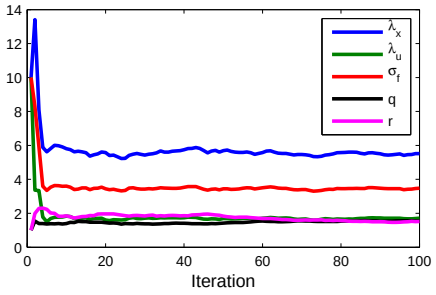


Figure 4.7: Convergence of parameters when learning a linear system using a squared exponential covariance function.

Figure 4.5 shows the convergence of the GP hyper-parameters (l_x and l_u) and noise parameters with respect to the PSAEM iteration. In order to judge the quality of the learnt GP-SSM we evaluate its predictive performance on the data set used for learning (training set) and on an independent data set generated from the same dynamical system (test set). The GP-SSM can make probabilistic predictions which report the uncertainty arising from the fact that only a finite amount of data is observed.

Figure 4.6 displays the predicted value of $\mathbf{f}_{t+1} - \mathbf{x}_t$ versus the true value. Recall that $\mathbf{f}_{t+1} - \mathbf{x}_t$ is equivalent to the step taken by the state in one single transition before process noise is added: $f(\mathbf{x}_t, \mathbf{u}_t) - \mathbf{x}_t$. One standard deviation error bars from the predictive distribution have also been plotted. Perfect predictions would lie on the unit slope line. We note that although the predictions are not perfect, error-bars tend to be large in predictions that are far from the true value and narrower for predictions that are closer to the truth. This is the desired outcome since the goal of the GP-SSM is to represent the uncertainty in its predictions.

We now move into a scenario in which the data is still generated by the linear dynamical system in (4.16) but we pretend that we are not aware of this linearity. In this case, a covariance function able to model nonlinear transition functions is a judicious choice. We use the squared exponential covariance function which imposes the assumption that the state transition function is smooth and infinitely differentiable (Rasmussen and Williams, 2006). Figure 4.7 shows, for a PSAEM run, the convergence of the covariance function

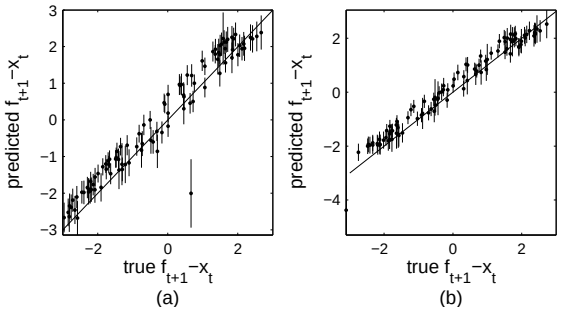


Figure 4.8: Linear dynamical system learnt using a GP-SSM with squared exponential covariance function. Predictions (a) on training data, and (b) on test data.

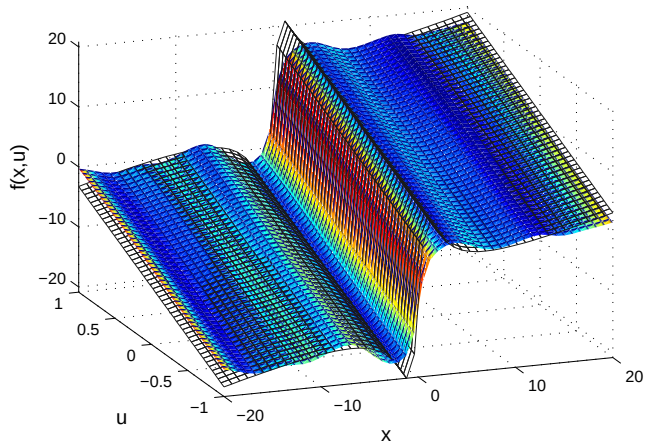


Figure 4.9: Nonlinear dynamical system with one state and one input. The black mesh represents the ground truth dynamics function and the colored surface is the mean of the identified function. Color is proportional to the standard deviation of the identified function (red represents high uncertainty and blue low uncertainty).

hyper-parameters (length-scales λ_x and λ_u and signal variance σ_f) and also the convergence of the noise variances.

The predictive performance on training data and independent test data is presented in Figure 4.8. Interestingly, in the panel corresponding to training data (a), there is particularly poor prediction that largely underestimates the value of the state transition. However, the variance for this prediction is very high which indicates that the identified model has little confidence in it. In this particular case, the mean of the prediction is 2.5 standard deviations away from the true value of the state transition.

4.3.3.2 LEARNING A NONLINEAR SYSTEM

GP-SSMs are particularly powerful for nonlinear system identification when it is not possible to create a parametric model of the system based on detailed knowledge about its dynamics. To illustrate this capability of GP-SSMs we use

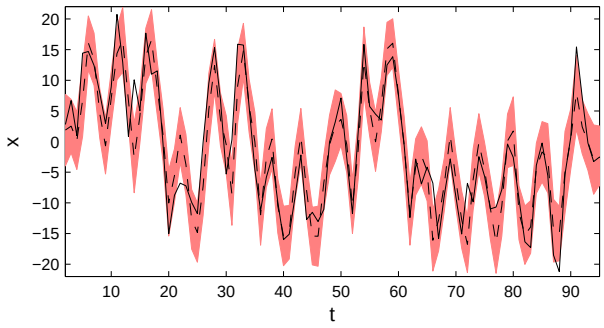


Figure 4.10: State trajectory from a test data set (solid black line). One step ahead predictions made with the identified model are depicted by a dashed line (mean) and a colored interval at ± 1 standard deviation (including process noise).

again the nonlinear dynamical system defined by

$$x_{t+1} = ax_t + b\frac{x_t}{1+x_t^2} + cu_t + v_t, \qquad v_t \sim \mathcal{N}(0, q), \tag{4.17a}$$

$$y_t = dx_t^2 + e_t, \qquad e_t \sim \mathcal{N}(0, r), \tag{4.17b}$$

with parameters $(a, b, c, d, q, r) = (0.5, 25, 8, 0.05, 10, 1)$ and a known input $u_t = \cos(1.2(t+1))$.

Again, we pretend that detailed knowledge about the particular form of (4.17a) is not available to us. We select a covariance function that consists of a Matérn covariance function in the $\mathbf{x}$ direction and a squared exponential in the $\mathbf{u}$ direction. The Matérn covariance function imposes less smoothness constraints than the squared exponential (Rasmussen and Williams, 2006) and is therefore more suited to model functions that can have sharp transitions.

Figure 4.9 shows the true state transition dynamics function (black mesh) and the identified function as a colored surface. Since the identified function from the GP-SSM comes in the form of a probability distribution over functions, the surface is plotted at $\mathbb{E}[\mathbf{f}_*|\mathbf{x}_*, \mathbf{u}_*, \mathbf{y}_{1:T}]$. The standard deviation of $\mathbf{f}_*$, which represents our uncertainty about the actual value of the function, is depicted by the color of the surface. Figure 4.10 shows the one step ahead predictive distributions $p(\mathbf{x}_{t+1}^*|\mathbf{x}_t^*, \mathbf{u}_t^*, \mathbf{y}_{0:T})$ on a test data set.

4.4 REDUCING THE COMPUTATIONAL COMPLEXITY

4.4.1 FIC COVARIANCE FUNCTION

The computational complexity of learning GP-SSMs can be reduced by using sparse covariance functions for the GP. In the following we present the use of the Fully Independent Conditional (FIC) covariance function first introduced in (Snelson and Ghahramani, 2006) and christened as such in (Quiñonero-Candela and Rasmussen, 2005). The model arising from the use of this particular covariance function does not need to be seen as an approximation to the "full" GP-SSM, it is still a GP-SSM, but with a *different prior over functions*. As a result,

it could even outperform its non-sparse version in the same way as it happens in some regression problems (Snelson and Ghahramani, 2006).

Most sparse GP methods can be formulated in terms of a set of inducing variables (Quiñonero-Candela and Rasmussen, 2005; Titsias, 2009). These variables live in the space of the latent function and have a set $\mathbf{z} \triangleq \{\mathbf{z}_i\}_{i=1}^M$ of corresponding inducing inputs. The assumption is that, conditionally on the inducing variables, the latent function values are mutually independent. Although the inducing variables are marginalised analytically, the inducing inputs have to be chosen in such a way that they, informally speaking, cover the same region of the input space covered by the data. Crucially, in order to achieve computational gains, the number M of inducing variables is selected to be smaller than the original number of data points.

As shown in (Quiñonero-Candela and Rasmussen, 2005), the FIC prior is obtained by replacing the covariance function $k(\cdot, \cdot)$ with

$$k^{\mathrm{FIC}}(\mathbf{x}_i, \mathbf{x}_j) = s(\mathbf{x}_i, \mathbf{x}_j) + \delta_{ij}\big(k(\mathbf{x}_i, \mathbf{x}_j) - s(\mathbf{x}_i, \mathbf{x}_j)\big), \tag{4.18}$$

where $s(\mathbf{x}_i, \mathbf{x}_j) \triangleq k(\mathbf{x}_i, \mathbf{z})k(\mathbf{z}, \mathbf{z})^{-1}k(\mathbf{z}, \mathbf{x}_j)$, δ_{ij} is Kronecker's delta and we use the convention whereby when $k(\cdot, \cdot)$ takes a set as one of its arguments it generates a matrix of covariances. Using the Woodbury matrix identity, we can express the one-step density $p(\mathbf{x}_t|\mathbf{x}_{0:t-1})$ of a GP-SSM, Equation (3.15b), as a Gaussian with the following mean and covariance matrix

$$\boldsymbol{\mu}_t^{\mathrm{FIC}}(\mathbf{x}_{0:t-1}) = \mathbf{m}_{t-1} + \mathbf{K}_{t-1,\mathbf{z}}\mathbf{P}\mathbf{K}_{\mathbf{z},0:t-2}\boldsymbol{\Lambda}_{0:t-2}^{-1}\left(\mathbf{x}_{1:t-1} - \mathbf{m}_{0:t-2}\right), \tag{4.19a}$$

$$\boldsymbol{\Sigma}_t^{\mathrm{FIC}}(\mathbf{x}_{0:t-1}) = \widetilde{\mathbf{K}}_{t-1} - \mathbf{S}_{t-1} + \mathbf{K}_{t-1,\mathbf{z}}\mathbf{P}\mathbf{K}_{\mathbf{z},t-1}, \tag{4.19b}$$

where

$$\mathbf{P} \triangleq (\mathbf{K}_{\mathbf{z}} + \mathbf{K}_{\mathbf{z},0:t-2}\boldsymbol{\Lambda}_{0:t-2}^{-1}\mathbf{K}_{0:t-2,\mathbf{z}})^{-1}, \tag{4.20a}$$

$$\boldsymbol{\Lambda}_{0:t-2} \triangleq \mathrm{diag}[\widetilde{\mathbf{K}}_{0:t-2} - \mathbf{S}_{0:t-2}], \tag{4.20b}$$

$$\mathbf{S}_{\mathcal{A},\mathcal{B}} \triangleq \mathbf{K}_{\mathcal{A},\mathbf{z}}\mathbf{K}_{\mathbf{z}}^{-1}\mathbf{K}_{\mathbf{z},\mathcal{B}}. \tag{4.20c}$$

Despite its apparent cumbersomeness, the computational complexity involved in computing the above mean and covariance is $\mathcal{O}(M^2 t)$. This contrasts with the $\mathcal{O}(t^3)$ complexity for general covariance functions when evaluating Equation (3.15b).

4.4.2 SEQUENTIAL CONSTRUCTION OF CHOLESKY FACTORISATIONS

A naive implementation of the CPF-AS algorithm gives rise to $\mathcal{O}(T^4)$ computational complexity, since at each time step $t = 1, ..., T$, a matrix of size $T \times T$ needs to be factorised. However, it is possible to update and reuse the factors from the previous time step, bringing the total computational complexity down to the familiar $\mathcal{O}(T^3)$. Furthermore, by introducing a sparse GP model,

we can reduce the complexity from $\mathcal{O}(M^2T^2)$ to $\mathcal{O}(M^2T)$ where $M \ll T$.

There are two costly operations of the CPF-AS algorithm: *(i)* sampling from the prior, requiring the computation of Equations (3.15c) and (3.15d) and *(ii)* evaluating the ancestor sampling probabilities which requires to evaluate the joint density $p(\mathbf{y}_{1:T}, \mathbf{x}_{0:T})$. Both of these operations can be carried out efficiently by keeping track of a Cholesky factorisation of the matrix

$$\widetilde{\mathbf{K}}(\{\mathbf{x}^i_{0:t-1}, \widetilde{\mathbf{x}}_{t:T-1}\}) = \mathbf{L}^i_t \mathbf{L}^{i\top}_t,$$

for each particle $i = 1, ..., N$ (see Section A.1.1 for the full PGAS algorithm). Here, $\widetilde{\mathbf{K}}(\{\mathbf{x}^i_{0:t-1}, \widetilde{\mathbf{x}}_{t:T-1}\})$ is a matrix defined analogously to $\widetilde{\mathbf{K}}_{0:T-1}$, but where the covariance function is evaluated for the concatenated state trajectory $\{\mathbf{x}^i_{0:t-1}, \widetilde{\mathbf{x}}_{t:T-1}\}$. From $\mathbf{L}^i_t$, it is possible to identify sub-matrices corresponding to the Cholesky factors for the covariance matrix $\mathbf{\Sigma}_t(\mathbf{x}^i_{0:t-1})$ as well as for the matrices needed to efficiently evaluate the ancestor sampling probabilities.

It remains to find an efficient update of the Cholesky factor to obtain $\mathbf{L}^i_{t+1}$. As we move from time t to $t+1$ in the algorithm, $\widetilde{\mathbf{x}}_t$ will be replaced by $\mathbf{x}^i_t$ in the concatenated trajectory. Hence, the matrix $\widetilde{\mathbf{K}}(\{\mathbf{x}^i_{0:t}, \widetilde{\mathbf{x}}_{t+1:T-1}\})$ can be obtained from $\widetilde{\mathbf{K}}(\{\mathbf{x}^i_{0:t-1}, \widetilde{\mathbf{x}}_{t:T-1}\})$ by replacing n_x rows and columns, corresponding to a rank $2n_x$ update. (Recall that n_x is the dimensionality of the state-space.) It follows that we can compute $\mathbf{L}^i_{t+1}$ by making n_x successive rank one updates and downdates on $\mathbf{L}^i_t$. Refer to (Gill et al., 1974) for a comprehensive review of methods for rank-one update/downdate of Cholesky factors and to the supplementary material of (Frigola et al., 2013) for a derivation tailored to the current problem.

In summary, for an arbitrary covariance function, all the operations at a specific time step can be done in $\mathcal{O}(T^2)$ computations, leading to a total computational complexity of $\mathcal{O}(T^3)$. For the GP-SSM with FIC covariance function, a naive implementation gives rise to $\mathcal{O}(M^2T^2)$ computational complexity. This can be reduced to $\mathcal{O}(M^2T)$ by keeping track of a factorisation for the matrix $\mathbf{P}$. However, to reach the $\mathcal{O}(M^2T)$ cost, all intermediate operations scaling with T have to be avoided, requiring the reuse of not only the matrix factorisations, but also the intermediate matrix-vector multiplications.

4.5 CONCLUSIONS

We have shown how to directly sample from the smoothing distribution of a GP-SSM without needing to know the nonlinear state transition function. Once samples from the smoothing distribution have been obtained, it is straightforward to describe the posterior over the state transition function.

We have also shown how samples from the smoothing distribution can be obtained using the Particle Gibbs with Ancestor Sampling algorithm (Lindsten et al., 2014). This algorithm has proven to achieve very good mixing while being extremely easy to use. Although the PGAS algorithm is able to effectively sample from very high-dimensional distributions for long time series (large T),

its reliance on a particle filter limits their effectiveness when the state dimension (n_x) is large.

The fully Bayesian learning approach presented in this chapter obtains joint samples from the smoothing distribution and the posterior distribution over the (hyper-)parameters. Therefore, uncertainty about all unknown variables is quantified. We have also presented an alternative empirical Bayes approach which can find maximum likelihood point estimate over all, or some, (hyper-)parameters. This can be useful for parameters over which it is hard to find a meaningful prior, e.g. the inducing inputs of a FIC covariance function.

Chapter 5

Gaussian Process State-Space Models – Variational Learning

In this chapter we introduce a Variational Inference approach to Bayesian learning of sparse Gaussian Process State-Space Models. The result of learning is a tractable approximate posterior over nonlinear dynamical systems that can provide very fast predictions at test time. In comparison to conventional parametric models this approach offers the possibility to straightforwardly trade off model capacity and computational cost whilst preventing overfitting.

As opposed to the learning algorithms presented in Chapter 4, the variational approach does not exactly marginalise out the nonlinear dynamics. Instead, it learns an explicit posterior of the dynamical system by computing an approximate posterior over the inducing variables of a sparse GP-SSM.

The presentation in this chapter expands the results previously published in (Frigola et al., 2014a).

5.1 Introduction

A GP-SSM has, in principle, the ability to model nonlinear state transition functions belonging to an infinite-dimensional space of functions. This contrasts with parametric nonlinear functions in which the *model capacity* is restricted a priori by the choice of the parametric form.

The sampling approaches introduced in Chapter 4 are able to learn the state transition function of a GP-SSM without the need of parametric approximations. However, they do so at a cost: each prediction of a state transition needs to average over samples of the latent state trajectory and hence has an $\mathcal{O}(TL)$ computational cost where T is the length of the time series and L is the number of samples.

The variational methods presented in this chapter will take a fundamentally different approach: they will attempt to represent the *posterior* over the state transition function with a finite number of parameters, the inducing points. The methodology shares many elements with the seminal work of Titsias (2009)

for Gaussian process regression but is adapted to the challenging setting of time series data. In a GP-SSM, the goal is to learn a function whose inputs and outputs are both unobserved.

Variational methods for GP-SSMs have two fundamental advantages over the Monte Carlo approaches presented in Chapter 4. First, they offer fast probabilistic predictions at test time with a computational complexity that is independent of the length of the time series. Second, mini-batch training is possible and leads to significant speed improvements when using large datasets. A more detailed discussion about advantages and disadvantages of variational inference is provided in Section 5.2.2.

5.2 EVIDENCE LOWER BOUND OF A GP-SSM

Variational inference is a popular method for approximate Bayesian inference based on making assumptions about the posterior over parameters and latent variables. These assumptions lead to a tractable lower bound on the marginal likelihood of the model. The marginal likelihood is also known as the evidence and its lower bound is sometimes referred to as the ELBO (Evidence Lower BOund). An approximation of the evidence is the objective function that we maximised for learning GP-SSMs in Section 4.3. In the variational inference setting, maximising the evidence lower bound is equivalent to minimising the Kullback-Leibler divergence between the approximate posterior and the exact one. (See Appendix A.2 for a very brief presentation and (Jordan et al., 1999) for an authoritative exposition.) Following standard variational inference methodology, we obtain the evidence lower bound of a GP-SSM augmented with inducing points $\mathbf{u}$ (Section 3.3)

$$\begin{aligned}\log p(\mathbf{y}|\boldsymbol{\theta}) &\geq \langle \log \frac{p(\mathbf{x}, \mathbf{f}, \mathbf{u}, \mathbf{y})}{q(\mathbf{x}, \mathbf{f}, \mathbf{u})} \rangle_{q(\mathbf{x},\mathbf{f},\mathbf{u})} \\ &= \langle \log \frac{p(\mathbf{u})p(\mathbf{x}_0)\prod_{t=1}^{T} p(\mathbf{f}_t|\mathbf{f}_{1:t-1}, \mathbf{x}_{0:t-1}, \mathbf{u})p(\mathbf{y}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{f}_t)}{q(\mathbf{x}, \mathbf{f}, \mathbf{u})} \rangle_{q(\mathbf{x},\mathbf{f},\mathbf{u})}.\end{aligned} \tag{5.1}$$

In order to make the formulas slightly less cumbersome, we are using the shorthand notation $\mathbf{x} \triangleq \mathbf{x}_{0:T}$, $\mathbf{y} \triangleq \mathbf{y}_{1:T}$, $\mathbf{f} \triangleq \mathbf{f}_{1:T}$, $\mathbf{u} \triangleq \mathbf{u}_{1:M}$, and $\langle f(x) \rangle_{p(x)} \triangleq \int f(x)p(x)\,\mathrm{d}x$.

At this point, it is critical to chose a variational distribution $q(\mathbf{x}, \mathbf{f}, \mathbf{u})$ that presents a favourable trade-off between tractability and expressivity. Inspired by a strand of work started by Titsias (2009), we chose a distribution for the latent variables in the GP-SSM that leads to cancellation of difficult terms involving the latent function. In particular, we use

$$q(\mathbf{x}, \mathbf{f}, \mathbf{u}) = q(\mathbf{u})q(\mathbf{x}) \prod_{t=1}^{T} p(\mathbf{f}_t|\mathbf{f}_{1:t-1}, \mathbf{x}_{0:t-1}, \mathbf{u}), \tag{5.2}$$

where $q(\mathbf{u})$ and $q(\mathbf{x})$ can take any form at this stage and will be variationally

optimised later. However, the terms relating to $\mathbf{f}$ are taken to match those of the prior. As a consequence, the difficult $p(\mathbf{f}_t|...)$ terms inside the log cancel out and lead to the following lower bound

$$\log p(\mathbf{y}|\boldsymbol{\theta}) \geq \langle \log \frac{p(\mathbf{u})p(\mathbf{x}_0)\prod_{t=1}^{T} \cancel{p(\mathbf{f}_t|\mathbf{f}_{1:t-1}, \mathbf{x}_{0:t-1}, \mathbf{u})}p(\mathbf{y}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{f}_t)}{q(\mathbf{u})q(\mathbf{x})\prod_{t=1}^{T} \cancel{p(\mathbf{f}_t|\mathbf{f}_{1:t-1}, \mathbf{x}_{0:t-1}, \mathbf{u})}} \rangle_{q(\mathbf{x},\mathbf{f},\mathbf{u})}, \tag{5.3a}$$

$$= \langle \log \frac{p(\mathbf{u})p(\mathbf{x}_0)\prod_{t=1}^{T} p(\mathbf{y}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{f}_t)}{q(\mathbf{u})q(\mathbf{x})} \rangle_{q(\mathbf{x},\mathbf{f},\mathbf{u})}, \tag{5.3b}$$

$$\triangleq \mathcal{L}(q(\mathbf{u}), q(\mathbf{x}), \boldsymbol{\theta}). \tag{5.3c}$$

Equation (5.3c) makes it explicit that the evidence lower bound is a function of the functions $q(\mathbf{u})$ and $q(\mathbf{x})$ and of the hyper-parameters $\boldsymbol{\theta}$. In the next section, we will use the calculus of variations to optimise the ELBO with respect to the functions $q(\mathbf{u})$ and $q(\mathbf{x})$. Now, we break down the ELBO into a sum of interpretable terms

$$\begin{aligned}
\mathcal{L}(q(\mathbf{u}), q(\mathbf{x}), \boldsymbol{\theta}) =& \langle \log \frac{p(\mathbf{u})}{q(\mathbf{u})} \rangle_{q(\mathbf{u})} + \langle \log \frac{1}{q(\mathbf{x})} \rangle_{q(\mathbf{x})} + \langle \log p(\mathbf{x}_0) \rangle_{q(\mathbf{x}_0)} \\
&+ \langle \log \prod_{t=1}^{T} p(\mathbf{x}_t|\mathbf{f}_t) \rangle_{q(\mathbf{x},\mathbf{f},\mathbf{u})} + \langle \log \prod_{t=1}^{T} p(\mathbf{y}_t|\mathbf{x}_t) \rangle_{q(\mathbf{x})} \\
=& -\mathrm{KL}(q(\mathbf{u})\|p(\mathbf{u})) + \mathcal{H}(q(\mathbf{x})) + \langle \log p(\mathbf{x}_0) \rangle_{q(\mathbf{x}_0)} \\
&+ \sum_{t=1}^{T} \left\{ \langle \underbrace{\langle \log p(\mathbf{x}_t|\mathbf{f}_t) \rangle_{p(\mathbf{f}_t|\mathbf{x}_{t-1},\mathbf{u})}}_{\Phi(\mathbf{x}_{t-1:t},\mathbf{u})} \rangle_{q(\mathbf{x})q(\mathbf{u})} + \langle \log p(\mathbf{y}_t|\mathbf{x}_t) \rangle_{q(\mathbf{x})} \right\},
\end{aligned} \tag{5.4}$$

where KL denotes the Kullback-Leibler divergence

$$\mathrm{KL}(q(\mathbf{u})\|p(\mathbf{u})) \triangleq \int q(\mathbf{u}) \log \frac{q(\mathbf{u})}{p(\mathbf{u})} \, \mathrm{d}\mathbf{u}, \tag{5.5}$$

$\mathcal{H}$ denotes the entropy

$$\mathcal{H}(q(\mathbf{x})) \triangleq -\int q(\mathbf{x}) \log q(\mathbf{x}) \, \mathrm{d}\mathbf{x}. \tag{5.6}$$

The integral with respect to $\mathbf{f}_t$ in the lower bound can be solved analytically[1]:

$$\Phi(\mathbf{x}_{t-1:t}, \mathbf{u}) = \langle \log p(\mathbf{x}_t|\mathbf{f}_t) \rangle_{p(\mathbf{f}_t|\mathbf{x}_{t-1},\mathbf{u})} \tag{5.7a}$$

$$= \langle \log \mathcal{N}(\mathbf{x}_t|\mathbf{f}_t, \mathbf{Q}) \rangle_{\mathcal{N}(\mathbf{f}_t|\mathbf{A}_{t-1}\mathbf{u},\mathbf{B}_{t-1})} \tag{5.7b}$$

$$= -\frac{1}{2}\mathrm{tr}(\mathbf{Q}^{-1}\mathbf{B}_{t-1}) + \log \mathcal{N}(\mathbf{x}_t|\mathbf{A}_{t-1}\mathbf{u}, \mathbf{Q}), \tag{5.7c}$$

[1] for simplicity of interpretation, the derivation that follows in the rest of the chapter is for the zero-mean GP case, see Section 5.7.3 for expressions for an arbitrary mean function

where

$$\mathbf{A}_{t-1} \triangleq \mathbf{K}_{t-1,\mathbf{z}}\mathbf{K}_{\mathbf{z}}^{-1}, \tag{5.8a}$$

$$\mathbf{B}_{t-1} \triangleq \mathbf{K}_{t-1,t-1} - \mathbf{K}_{t-1,\mathbf{z}}\mathbf{K}_{\mathbf{z}}^{-1}\mathbf{K}_{\mathbf{z},t-1}. \tag{5.8b}$$

As we will see in Section 5.3.1, the distribution $q(\mathbf{u})$ that maximises the ELBO is a Gaussian distribution: $\mathcal{N}(\mathbf{u}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$. Therefore, the following expectation in the ELBO has a closed form expression

$$\langle \Phi(\mathbf{x}_{t-1:t}, \mathbf{u}) \rangle_{q(\mathbf{u})} = \int \Phi(\mathbf{x}_{t-1:t}, \mathbf{u}) \mathcal{N}(\mathbf{u}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) \, \mathrm{d}\mathbf{u} \tag{5.9a}$$

$$= -\frac{1}{2}\mathrm{tr}(\mathbf{Q}^{-1}(\mathbf{B}_{t-1} + \mathbf{A}_{t-1}\boldsymbol{\Sigma}\mathbf{A}_{t-1}^{\top})) + \log \mathcal{N}(\mathbf{x}_t|\mathbf{A}_{t-1}\boldsymbol{\mu}, \mathbf{Q}). \tag{5.9b}$$

Therefore, the ELBO becomes

$$\begin{aligned}\mathcal{L}(q(\mathbf{u}), q(\mathbf{x}), \boldsymbol{\theta}) = &- \mathrm{KL}(q(\mathbf{u})\|p(\mathbf{u})) + \mathcal{H}(q(\mathbf{x})) + \langle \log p(\mathbf{x}_0) \rangle_{q(\mathbf{x}_0)} \\ &+ \sum_{t=1}^{T} \Big\{ -\frac{1}{2}\langle \mathrm{tr}(\mathbf{Q}^{-1}(\mathbf{B}_{t-1} + \mathbf{A}_{t-1}\boldsymbol{\Sigma}\mathbf{A}_{t-1}^{\top})) \rangle_{q(\mathbf{x}_{t-1})} \\ &+ \langle \log \mathcal{N}(\mathbf{x}_t|\mathbf{A}_{t-1}\boldsymbol{\mu}, \mathbf{Q}) \rangle_{q(\mathbf{x}_{t-1:t})} \\ &+ \langle \log p(\mathbf{y}_t|\mathbf{x}_t) \rangle_{q(\mathbf{x}_t)} \Big\}. \end{aligned} \tag{5.10}$$

5.2.1 INTERPRETATION OF THE LOWER BOUND

We can provide an interpretation of the influence that each term in Equation (5.10) has when maximising the evidence lower bound

- $-\mathrm{KL}(q(\mathbf{u})\|p(\mathbf{u}))$: regularisation term encouraging the approximate posterior $q(\mathbf{u})$ to remain close to the prior $p(\mathbf{u})$ which encodes assumptions (e.g. smoothness) about the state transition function.

- $\mathcal{H}(q(\mathbf{x}))$: entropy term discouraging overly tight smoothing distributions over the state trajectories.

- $\langle \log p(\mathbf{x}_0) \rangle_{q(\mathbf{x}_0)}$: encourages the state trajectory to start in a region of the state space where the prior is high. This term can be neglected if the prior on the initial state is sufficiently broad.

- $-\frac{1}{2}\langle \mathrm{tr}(\mathbf{Q}^{-1}\mathbf{B}_{t-1}) \rangle_{q(\mathbf{x}_{t-1})}$: penalises states that are far from the inducing inputs. This is apparent when interpreting $\mathbf{B}_{t-1}$ as the predictive covariance of GP regression at a test point $\mathbf{x}_{t-1}$ with $\mathbf{z}$ as the regression inputs (5.8b). States close to the inducing inputs have little predictive variance and result in a small penalty.

- $-\frac{1}{2}\langle \mathrm{tr}(\mathbf{Q}^{-1}\mathbf{A}_{t-1}\boldsymbol{\Sigma}\mathbf{A}_{t-1}^{\top}) \rangle_{q(\mathbf{x}_{t-1})}$: penalises variance in the approximate posterior $q(\mathbf{u})$. It opposes the KL term between prior and posterior over

$\mathbf{u}$. The expression $\mathbf{A}_{t-1}\boldsymbol{\Sigma}\mathbf{A}_{t-1}^{\top}$ can be interpreted as the extra covariance in a prediction of $\mathbf{f}_t$ due to variance in the posterior of $\mathbf{u}$, see Equation (5.27d).

- $\langle \log \mathcal{N}(\mathbf{x}_t|\mathbf{A}_{t-1}\boldsymbol{\mu}, \mathbf{Q})\rangle_{q(\mathbf{x}_{t-1:t})}$: crucial term that quantifies the agreement between neighbouring states $(\mathbf{x}_t, \mathbf{x}_{t-1})$ and the mean of the (approximate) posterior over the inducing points of the state transition function.

- $\langle \log p(\mathbf{y}_t|\mathbf{x}_t)\rangle_{q(\mathbf{x}_t)}$: data fit term encouraging state trajectories that give high probability to the observed data.

5.2.2 PROPERTIES OF THE LOWER BOUND

As in other sparse GP models, variational inference gives the ability to obtain an approximate posterior of the state transition function of a GP-SSM at the locations of the inducing inputs. Away from those locations, the posterior takes the form of the prior conditioned on the inducing variables. By increasing the number of inducing variables, the ELBO can only become tighter (as in (Titsias, 2009)). This offers a straightforward trade-off between model capacity and computation cost without increasing the risk of overfitting.

The evidence lower bound provided by variational inference is very useful to obtain point estimates of hyper-parameters that approximate the maximum likelihood solution. Also, in many applications it has been found that maximising the ELBO results in shorter training times compared to Monte Carlo-based methods.

However, variational inference is based on making assumptions about the shape of the approximate posterior such as factorisation assumptions or specifying a particular parametric form for the approximate posterior. This contrasts with Monte Carlo methods where no such assumptions are made.

As useful as variational inference is, it is important to keep in mind the characteristics of the approximation. Perhaps the most important characteristic is that approximations from variational inference tend to be more compact than the true distribution (MacKay, 2003) although this is not strictly always the case (Turner and Sahani, 2011). Underestimating uncertainty is an undesirable property because it gives an illusion of certainty. However, as MacKay (2003) argues, it is often better to underestimate uncertainty rather than to ignore it altogether.

Another issue with variational inference is due to the fact that maximising the evidence lower bound biases the solution towards areas where the bound is tighter (Turner and Sahani, 2011). As a consequence, the optimal setting of the parameters on the lower bound is not necessarily the maximum of the evidence. The bias in the parameters will depend on the particular characteristics of the lower bound that we chose to maximise.

5.2.3 ARE THE INDUCING INPUTS VARIATIONAL PARAMETERS?

The inducing inputs $\mathbf{z}$ appear both in the generative model of a sparse GP-SSM, Equation (3.29), and in the variational distribution, Equation. (5.2). It could appear that they are parameters of the model that are being reused in the variational distribution much in the same way as the covariance function hyper-parameters. However, on close observation it is apparent that the inducing inputs in a sparse GP-SSM behave in a very particular manner: if the inducing *variables* are integrated out, the marginal distribution over the rest of variables (i.e. the non-sparse GP-SSM) does not depend on the inducing *inputs*

$$\int p(\mathbf{y}, \mathbf{x}, \mathbf{f}, \mathbf{u} \mid \mathbf{z}) \, \mathrm{d}\mathbf{u} = p(\mathbf{y}, \mathbf{x}, \mathbf{f} \mid \mathbf{z}) = p(\mathbf{y}, \mathbf{x}, \mathbf{f}). \tag{5.11}$$

Since the location of the inducing inputs does not affect the distribution over observed data, state trajectories or latent function, the inducing inputs become parameters that can be tuned at our will. Adjusting the inducing inputs does not change our assumptions about how the data was generated. But, crucially, adjusting the inducing inputs changes the variational distribution $q(\mathbf{x}, \mathbf{f}, \mathbf{u})$ and can lead to a tighter lower bound on the model evidence.

5.3 OPTIMAL VARIATIONAL DISTRIBUTIONS

5.3.1 OPTIMAL VARIATIONAL DISTRIBUTION FOR $\mathbf{u}$

The optimal distribution for $q(\mathbf{u})$ is a stationary point of the functional $\mathcal{L}(q(\mathbf{u}), q(\mathbf{x}), \boldsymbol{\theta})$. The evidence lower bound has a stationary point with respect to the function $q(\mathbf{u})$ if and only if this function satisfies the Euler-Lagrange equation (Appendix A.2)

$$\frac{\partial}{\partial q(\mathbf{u})} \left\{ q(\mathbf{u}) \log \frac{p(\mathbf{u})}{q(\mathbf{u})} + q(\mathbf{u}) \sum_{t=1}^{T} \langle \Phi(\mathbf{x}_{t-1:t}, \mathbf{u}) \rangle_{q(\mathbf{x}_{t-1:t})} \right\} = 0, \tag{5.12}$$

which results in an optimal distribution $q^*(\mathbf{u})$

$$0 = \log \frac{p(\mathbf{u})}{q^*(\mathbf{u})} - 1 + \sum_{t=1}^{T} \langle \Phi(\mathbf{x}_{t-1:t}, \mathbf{u}) \rangle_{q(\mathbf{x}_{t-1:t})}, \tag{5.13a}$$

$$\log q^*(\mathbf{u}) = \log p(\mathbf{u}) - 1 + \sum_{t=1}^{T} \langle \Phi(\mathbf{x}_{t-1:t}, \mathbf{u}) \rangle_{q(\mathbf{x}_{t-1:t})}, \tag{5.13b}$$

which, after exponentiation, becomes

$$q^*(\mathbf{u}) \propto p(\mathbf{u}) \prod_{t=1}^{T} \exp \langle \log \mathcal{N}(\mathbf{x}_t | \mathbf{A}_{t-1}\mathbf{u}, \mathbf{Q}) \rangle_{q(\mathbf{x}_{t-1:t})}. \tag{5.14}$$

In the following, we show that the optimal variational distribution $q^*(\mathbf{u})$ is, conveniently, a multivariate Gaussian distribution. This is not an extra as-

sumption that we have added to the variational inference but a natural consequence of the evidence lower bound that is being maximised.

$$\begin{aligned} q^*(\mathbf{u}) &\propto \mathcal{N}(\mathbf{u} \mid \mathbf{0}, \mathbf{K}_\mathbf{z}) \prod_{t=1}^{T} \exp\langle \log \mathcal{N}(\mathbf{x}_t | \mathbf{A}_{t-1}\mathbf{u}, \mathbf{Q}) \rangle_{q(\mathbf{x}_{t-1:t})}, \\ &\propto \mathcal{N}(\mathbf{u} \mid \mathbf{0}, \mathbf{K}_\mathbf{z}) \prod_{t=1}^{T} \exp\langle \mathbf{x}_t^T \mathbf{Q}^{-1} \mathbf{A}_{t-1}\mathbf{u} - \frac{1}{2}\mathbf{u}^T \mathbf{A}_{t-1}^T \mathbf{Q}^{-1} \mathbf{A}_{t-1}\mathbf{u} \rangle_{q(\mathbf{x}_{t-1:t})}, \\ &\propto \exp\{ \sum_{t=1}^{T} \langle \mathbf{x}_t^T \mathbf{Q}^{-1} \mathbf{A}_{t-1} \rangle_{q(\mathbf{x}_{t-1:t})} \mathbf{u} \\ &\quad - \frac{1}{2}\mathbf{u}^T \left(\mathbf{K}_\mathbf{z}^{-1} + \sum_{t=1}^{T} \langle \mathbf{A}_{t-1}^T \mathbf{Q}^{-1} \mathbf{A}_{t-1} \rangle_{q(\mathbf{x}_{t-1})} \right) \mathbf{u} \}. \end{aligned} \tag{5.15a}$$

From this expression we confirm that $q^*(\mathbf{u})$ is indeed a Gaussian distribution with natural parameters

$$\boldsymbol{\eta}_1 = \sum_{t=1}^{T} \langle \mathbf{A}_{t-1}^T \mathbf{Q}^{-1} \mathbf{x}_t \rangle_{q(\mathbf{x}_{t-1:t})}, \tag{5.16a}$$

$$\boldsymbol{\eta}_2 = -\frac{1}{2} \left(\mathbf{K}_\mathbf{z}^{-1} + \sum_{t=1}^{T} \langle \mathbf{A}_{t-1}^T \mathbf{Q}^{-1} \mathbf{A}_{t-1} \rangle_{q(\mathbf{x}_{t-1})} \right). \tag{5.16b}$$

Therefore, the optimal distribution $q^*(\mathbf{u})$ depends on the sufficient statistics

$$\boldsymbol{\Psi}_1 \triangleq \sum_{t=1}^{T} \langle \mathbf{K}_{t-1,\mathbf{z}}^T \mathbf{Q}^{-1} \mathbf{x}_t \rangle_{q(\mathbf{x}_{t-1:t})}, \tag{5.17a}$$

$$\boldsymbol{\Psi}_2 \triangleq \sum_{t=1}^{T} \langle \mathbf{K}_{t-1,\mathbf{z}}^T \mathbf{Q}^{-1} \mathbf{K}_{t-1,\mathbf{z}} \rangle_{q(\mathbf{x}_{t-1})}. \tag{5.17b}$$

The mean and covariance matrix of $q^*(\mathbf{u})$, denoted as $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ respectively, can be computed as $\boldsymbol{\mu} = \boldsymbol{\Sigma}\boldsymbol{\eta}_1$ and $\boldsymbol{\Sigma} = (-2\boldsymbol{\eta}_2)^{-1}$

$$\boldsymbol{\mu} = \boldsymbol{\Sigma}\mathbf{K}_\mathbf{z}^{-1}\boldsymbol{\Psi}_1 = \left(\mathbf{I} + \mathbf{K}_\mathbf{z}^{-1}\boldsymbol{\Psi}_2 \right)^{-1} \boldsymbol{\Psi}_1, \tag{5.18a}$$

$$\boldsymbol{\Sigma} = \left(\mathbf{K}_\mathbf{z}^{-1} + \mathbf{K}_\mathbf{z}^{-1}\boldsymbol{\Psi}_2\mathbf{K}_\mathbf{z}^{-1} \right)^{-1}. \tag{5.18b}$$

5.3.2 OPTIMAL VARIATIONAL DISTRIBUTION FOR $\mathbf{x}$

We follow an analogous procedure to find the optimal distribution over the state trajectory that maximises the evidence lower bound for a fixed $q(\mathbf{u})$. The Euler-Lagrange equation with respect to the distribution $q(\mathbf{x})$ is

$$\begin{aligned} \frac{\partial}{\partial q(\mathbf{x})} \big\{ -q(\mathbf{x}) \log q(\mathbf{x}) + q(\mathbf{x}_0) \log p(\mathbf{x}_0) + \sum_{t=1}^{T} q(\mathbf{x}) \langle \Phi(\mathbf{x}_{t-1:t}, \mathbf{u}) \rangle_{q(\mathbf{u})} \\ + \sum_{t=1}^{T} q(\mathbf{x}) \log p(\mathbf{y}_t | \mathbf{x}_t) \big\} = 0, \end{aligned} \tag{5.19}$$

which results in an optimal distribution $q^*(\mathbf{x})$

$$0 = -\log q^*(\mathbf{x}) - 1 + \log p(\mathbf{x}_0) + \sum_{t=1}^{T} \big\{ \langle \Phi(\mathbf{x}_{t-1:t}, \mathbf{u}) \rangle_{q(\mathbf{u})} + \log p(\mathbf{y}_t | \mathbf{x}_t) \big\}, \quad (5.20)$$

which, after exponentiation and using $q(\mathbf{u}) = \mathcal{N}(\mathbf{u}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$, becomes

$$q^*(\mathbf{x}) \propto p(\mathbf{x}_0) \prod_{t=1}^{T} p(\mathbf{y}_t|\mathbf{x}_t) \exp\{-\frac{1}{2}\mathrm{tr}\big(\mathbf{Q}^{-1}(\mathbf{B}_{t-1}+\mathbf{A}_{t-1}\boldsymbol{\Sigma}\mathbf{A}_{t-1}^T)\big)\} \mathcal{N}(\mathbf{x}_t|\mathbf{A}_{t-1}\boldsymbol{\mu}, \mathbf{Q}). \quad (5.21)$$

Examining the optimal distribution $q^*(\mathbf{x})$, we can see that it can be interpreted as the smoothing distribution of an *auxiliary* parametric state-space model. This auxiliary model is simpler than a GP-SSM since the latent states factorise with a *Markovian* structure. Equation (5.21) can be interpreted as a the joint distribution of a nonlinear state-space model with a Gaussian state transition density

$$\tilde{p}(\mathbf{x}_t \mid \mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t|\mathbf{A}_{t-1}\boldsymbol{\mu}, \mathbf{Q}), \quad (5.22)$$

and a likelihood augmented with an additional term

$$\tilde{p}(\mathbf{y} \mid \mathbf{x}) \propto \prod_{t=1}^{T} p(\mathbf{y}_t|\mathbf{x}_t) \exp\{-\frac{1}{2}\mathrm{tr}\big(\mathbf{Q}^{-1}(\mathbf{B}_{t-1} + \mathbf{A}_{t-1}\boldsymbol{\Sigma}\mathbf{A}_{t-1}^T)\big)\}. \quad (5.23)$$

Recall that $\mathbf{A}_{t-1}$ and $\mathbf{B}_{t-1}$ depend on $\mathbf{x}_{t-1}$.

To gain an intuition about the additional term $\exp\{...\}$, it is useful to consider the case where there is no uncertainty about $\mathbf{u}$ and, therefore, the additional term becomes $\exp\{-\frac{1}{2}\mathrm{tr}\big(\mathbf{Q}^{-1}\mathbf{B}_{t-1}\big)\}$. $\mathbf{B}_{t-1}$ is the predictive variance of Gaussian process regression at a test point $\mathbf{x}_{t-1}$ when using the inducing points and inducing inputs as noiseless training data. Therefore, the predictive variance is zero if $\mathbf{x}_{t-1}$ has the same value as one of the inducing inputs. The additional likelihood term in Equation (5.23) becomes larger when the states are closer to the inducing inputs.

Smoothing in a nonlinear Markovian state-space model such as this auxiliary model is a standard problem in the context of time series. There exist many strategies to find the smoothing distribution (Särkkä, 2013). However, their performance varies depending on the characteristics of the peculiarities of the problem. For instance, in a mildly nonlinear system with Gaussian noise, an extended Kalman smoother can have good performance by making a Gaussian approximation to the smoothing distribution. On the other hand, problems with severe nonlinearities and/or non-Gaussian likelihoods can lead to heavily multimodal smoothing distributions that are better represented using particle methods such as sequential Monte Carlo (SMC) or Particle Markov Chain Monte Carlo (PMCMC) (Andrieu et al., 2010).

Algorithm 3 Variational learning of GP-SSMs with particle smoothing and vanilla gradient descent. Batch mode (i.e. non-SVI) is the particular case where the mini-batch is the whole dataset.

Require: Observations $\mathbf{y}_{1:T}$. Initial values for $\boldsymbol{\theta}, \boldsymbol{\eta}_1$ and $\boldsymbol{\eta}_2$. Schedules for ρ and λ. $i = 1$.
repeat
  $\mathbf{y}_{\tau:\tau'} \leftarrow \text{SAMPLEMINIBATCH}(\mathbf{y}_{1:T})$
  $\{\mathbf{x}_{\tau:\tau'}\}_{l=1}^{L} \leftarrow \text{GETSAMPLESOPTIMALQX}(\mathbf{y}_{\tau:\tau'}, \boldsymbol{\theta}, \boldsymbol{\eta}_1, \boldsymbol{\eta}_2)$ sample eq. (5.21)
  $\nabla_{\boldsymbol{\theta}} \mathcal{L} \leftarrow \text{GETTHETAGRADIENT}(\{\mathbf{x}_{\tau:\tau'}\}_{l=1}^{L}, \boldsymbol{\theta})$ supp. material
  $\boldsymbol{\eta}_1^*, \boldsymbol{\eta}_2^* \leftarrow \text{GETOPTIMALQU}(\{\mathbf{x}_{\tau:\tau'}\}_{l=1}^{L}, \boldsymbol{\theta})$ eq. (5.16) or (5.28)
  $\boldsymbol{\eta}_1 \leftarrow \boldsymbol{\eta}_1 + \rho_i(\boldsymbol{\eta}_1^* - \boldsymbol{\eta}_1)$
  $\boldsymbol{\eta}_2 \leftarrow \boldsymbol{\eta}_2 + \rho_i(\boldsymbol{\eta}_2^* - \boldsymbol{\eta}_2)$
  $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \lambda_i \nabla_{\boldsymbol{\theta}} \mathcal{L}$
  $i \leftarrow i + 1$
until ELBO convergence.

5.4 OPTIMISING THE EVIDENCE LOWER BOUND

To learn the model, we maximise the evidence lower bound with respect to the variational distributions and the hyper-parameters. We employ a procedure that alternates between sampling from the optimal distribution $q^*(\mathbf{x})$, updating the natural parameters of $q^*(\mathbf{u})$ and applying gradient ascent in $\boldsymbol{\theta}$.

Algorithm 3 presents a simple gradient ascent strategy with only two learning rates. The optimisation could also be carried out using more advanced stochastic optimisation techniques with additional learning rates and strategies to set them, e.g. AdaGrad (Duchi et al., 2011), RMSProp (Hinton, 2012), Adam (Kingma and Ba, 2015), etc.

As discussed in Section 5.2.3, the inducing inputs $\mathbf{z}$ can be considered variational parameters. Therefore, as far as the optimisation of the lower bound is concerned, we can bundle the inducing inputs together with the rest of the model hyper-parameters $\boldsymbol{\theta}$ and optimise them jointly.

5.4.1 ALTERNATIVE OPTIMISATION STRATEGY

In an analogous way to (Titsias, 2009), it is possible to plug the optimal $q^*(\mathbf{u})$ into the lower bound to obtain a version of the lower bound that depends exclusively on $q(\mathbf{x})$ and $\boldsymbol{\theta}$

$$\begin{aligned}\tilde{\mathcal{L}}(q(\mathbf{x}), \boldsymbol{\theta}) = &- \mathrm{KL}(q^*(\mathbf{u}) \| p(\mathbf{u})) + \mathcal{H}(q(\mathbf{x})) + \langle \log p(\mathbf{x}_0) \rangle_{q(\mathbf{x}_0)} \\ &+ \sum_{t=1}^{T} \left\{ \langle \Phi(\mathbf{x}_{t-1:t}, \mathbf{u}) \rangle_{q^*(\mathbf{u})q(\mathbf{x})} + \langle \log p(\mathbf{y}_t | \mathbf{x}_t) \rangle_{q(\mathbf{x})} \right\}, \end{aligned} \tag{5.24}$$

where $q^*(\mathbf{u}) = \mathcal{N}(\mathbf{u} \mid \boldsymbol{\mu}(q(\mathbf{x})), \boldsymbol{\Sigma}(q(\mathbf{x})))$ uses the means and covariances in Equation (5.18) which are a function of $q(\mathbf{x})$.

Finding an explicit expression for the distribution $q^*(\mathbf{x})$ that maximises $\tilde{\mathcal{L}}$ would be extremely useful. It would avoid the need to take steps in the natural parameters of $q^*(\mathbf{u})$ as proposed in Algorithm 3. In an analogous way as in Section 5.3.2, we proceed to find a stationary point of $\tilde{\mathcal{L}}$ with respect to $q(\mathbf{x})$

using the Euler-Lagrange equation. However, now there is an added difficulty since the term $\langle \Phi(\mathbf{x}_{t-1:t}, \mathbf{u}) \rangle_{q^*(\mathbf{u})}$ depends on $q(\mathbf{x})$

$$\begin{aligned} 0 =& \frac{\partial}{\partial q(\mathbf{x})} \big\{ -q(\mathbf{x}) \log q(\mathbf{x}) + q(\mathbf{x}_0) \log p(\mathbf{x}_0) \\ &+ \sum_{t=1}^{T} q(\mathbf{x}) \langle \Phi(\mathbf{x}_{t-1:t}, \mathbf{u}, q(\mathbf{x})) \rangle_{q^*(\mathbf{u})} \\ &+ \sum_{t=1}^{T} q(\mathbf{x}) \log p(\mathbf{y}_t | \mathbf{x}_t) \big\}. \end{aligned} \tag{5.25}$$

Which results in

$$\begin{aligned} 0 =& - \log q^*(\mathbf{x}) - 1 + \log p(\mathbf{x}_0) \\ &+ \sum_{t=1}^{T} \big\{ \langle \Phi(\mathbf{x}_{t-1:t}, \mathbf{u}, q^*(\mathbf{x})) \rangle_{q^*(\mathbf{u})} \\ &+ q^*(\mathbf{x}) \frac{\partial}{\partial q(\mathbf{x})} \langle \Phi(\mathbf{x}_{t-1:t}, \mathbf{u}, q(\mathbf{x})) \rangle_{q^*(\mathbf{u})} \big|_{q(\mathbf{x})=q^*(\mathbf{x})} \\ &+ \log p(\mathbf{y}_t | \mathbf{x}_t) \big\}. \end{aligned} \tag{5.26}$$

Developing this equation further leads to expectations over products of kernel matrices that may be analytically tractable for some choices of the covariance function. However, I have so far been unable to solve them.

5.5 MAKING PREDICTIONS

One of the most appealing properties of the variational approach to learning GP-SSMs is that the approximate predictive distribution of the state transition function at a test point $\mathbf{x}_*$ can be cheaply computed

$$p(\mathbf{f}_* | \mathbf{x}_*, \mathbf{y}) = \int p(\mathbf{f}_* | \mathbf{x}_*, \mathbf{x}, \mathbf{u})\, p(\mathbf{x} | \mathbf{u}, \mathbf{y})\, p(\mathbf{u} | \mathbf{y})\, \mathrm{d}\mathbf{x}\mathrm{d}\mathbf{u} \tag{5.27a}$$

$$\approx \int p(\mathbf{f}_* | \mathbf{x}_*, \mathbf{u})\, p(\mathbf{x} | \mathbf{u}, \mathbf{y})\, q(\mathbf{u})\, \mathrm{d}\mathbf{x}\mathrm{d}\mathbf{u} \tag{5.27b}$$

$$= \int p(\mathbf{f}_* | \mathbf{x}_*, \mathbf{u})\, q(\mathbf{u})\, \mathrm{d}\mathbf{u} \tag{5.27c}$$

$$= \mathcal{N}(\mathbf{f}_* | \mathbf{A}_* \boldsymbol{\mu}, \mathbf{B}_* + \mathbf{A}_* \boldsymbol{\Sigma} \mathbf{A}_*^\top). \tag{5.27d}$$

This derivation contains two approximations: 1) predictions at new test points are considered to depend only on the inducing variables, and 2) the posterior distribution over $\mathbf{u}$ is approximated by a variational distribution (Section 5.3.1).

The predictive covariance can be decomposed in two terms. The term $\mathbf{B}_*$ can be interpreted as the variance due to the fact that there are only a finite number of inducing points. An infinite amount of inducing inputs spread over the state space would result in $\mathbf{B}_* = \mathbf{0}$ for any $\mathbf{x}_*$. On the other hand, the term $\mathbf{A}_* \boldsymbol{\Sigma} \mathbf{A}_*^\top$ is a consequence of variance in the posterior over the inducing points. Perfect knowledge of $\mathbf{u}$ would result in $\mathbf{A}_* \boldsymbol{\Sigma} \mathbf{A}_*^\top = \mathbf{0}$.

After pre-computations, the cost of each prediction is $\mathcal{O}(M)$ for the mean and $\mathcal{O}(M^2)$ for the variance. This contrasts with the $\mathcal{O}(TL)$ and $\mathcal{O}(T^2L)$ complexity of approaches based on sampling from the smoothing distribution where the predictive distribution is approximated with L samples from $p(\mathbf{x}|\mathbf{y})$ (Chapter 4). Instead, the variational approach condenses the learning of the latent function on the inducing points $\mathbf{u}$ and does not explicitly need the smoothing distribution $p(\mathbf{x}|\mathbf{y})$ to make predictions.

5.6 EXTENSIONS

5.6.1 STOCHASTIC VARIATIONAL INFERENCE

When applying conventional variational inference to long time series, it is necessary to draw samples of state trajectories from $q^*(\mathbf{x})$ which have the same length as the time series. This computationally costly operation needs to be re-run every time that the parameters are updated. Such a strategy is particularly inefficient at the start of the optimisation process, when the parameters are far from optimal. In this situation, samples are drawn from $q^*(\mathbf{x})$ using parameters that misrepresent the system's dynamics and, as a consequence, produce poor state trajectories. *Stochastic variational inference* (SVI) techniques have been developed to mitigate this problem by using approximate gradients/updates which can be computed much faster than the conventional ones. This is achieved by using only a subset of data at each step instead of the full dataset (Neal and Hinton, 1998; Hoffman et al., 2013). As a result, many steps in parameter space can be taken in the time that would be required to make a single step using the full dataset. This enables swift progress in parameter space by reducing wasted computation in areas that are clearly not optimal.

Stochastic variational inference techniques usually make the assumption that all data points are independent. This allows for the computation of unbiased estimates of expectations using only subsets of the data selected at random from the full data set. Those subsets of data are often known as *mini-batches*. In a time series context, data points are not independent and their order is of crucial importance. Therefore, it is necessary to develop a customised version of SVI.

In our formulation, both $q^*(\mathbf{u})$ and $\frac{\partial \mathcal{L}}{\partial \boldsymbol{\theta}}$ depend on $q(\mathbf{x})$ via sufficient statistics that contain a summation over all elements in the state trajectory. In particular, these sufficient statistics depend only on marginals involving states at a single time step $q(\mathbf{x}_t)$ or at two neighbouring time steps $q(\mathbf{x}_{t-1:t})$. We can obtain *unbiased* estimates of the sufficient statistics by using segments of the sequence that are sampled uniformly at random. Estimates of the optimal parameters of $q^*(\mathbf{u})$ can be computed using only a segment of the time series of

length S spanning from time τ to τ'

$$\boldsymbol{\eta}_1 \approx \frac{T}{S}\sum_{t=\tau}^{\tau'}\langle \mathbf{A}_{t-1}^T \mathbf{Q}^{-1}\mathbf{x}_t\rangle_{q(\mathbf{x}_{t-1:t})}, \tag{5.28a}$$

$$\boldsymbol{\eta}_2 \approx -\frac{1}{2}\left(\mathbf{K}_{\mathbf{z}}^{-1} + \frac{T}{S}\sum_{t=\tau}^{\tau'}\langle \mathbf{A}_{t-1}^T \mathbf{Q}^{-1}\mathbf{A}_{t-1}\rangle_{q(\mathbf{x}_{t-1})}\right). \tag{5.28b}$$

In order to obtain unbiased estimates, the only requirement is that segments are sampled in such a way that all terms in the sum from $t = 1$ to T have the same probability to appear in the sample.

Although we can obtain unbiased estimates that only involve sums over a part of the time series, it is still necessary to compute $q^*(\mathbf{x}_{\tau-1:\tau'})$. Such an operation has complexity $\mathcal{O}(T)$ because the full observed time series is necessary to obtain the smoothing distribution. That would negate most of the computational benefit offered by SVI. However, in practice, $q^*(\mathbf{x}_{\tau-1:\tau'})$ can be well approximated by running the smoothing algorithm locally around the segment $\tau - 1 : \tau$. This can be justified by the fact that the smoothing distribution at a particular time is less affected by measurements that are far into the past or into the future (Särkkä, 2013). In parallel with our work (Frigola et al., 2014a), Foti et al. (2014) presented the use of stochastic variational inference for Hidden Markov Models (HMM). Foti et al. (2014) compute an approximation to $q^*(\mathbf{x}_{\tau-1:\tau'})$ by using observations beyond the edges of the sequence: $\mathbf{y}_{\tau-1-l:\tau'+l}$. They also provide an algorithm to adaptively change l although they report experimental results showing that $l << S$ is sufficient unless S is very small.

Finally, we note that our stochastic variational inference algorithm is an instance of *doubly* stochastic variational inference (Titsias and Lázaro-Gredilla, 2014). One source of stochasticity originates from the use of mini-batches of data at each time step. The second source originates from the computation of expectations with respect to $q^*(\mathbf{x})$ that are approximated using samples from that distribution.

5.6.2 ONLINE LEARNING

Our variational approach to learn GP-SSMs also leads naturally to an online learning implementation. This is of particular interest in the context of dynamical systems as it is often the case that data arrives in a sequential manner, e.g. a robot learning the dynamics of different objects by interacting with them. Online learning in a Bayesian setting consists in sequential application of Bayes rule whereby the posterior after observing data up to time t becomes the prior at time $t + 1$ (Opper, 1998; Broderick et al., 2013). In our case, this involves replacing the prior $p(\mathbf{u}) = \mathcal{N}(\mathbf{u}|\mathbf{0}, \mathbf{K}_{\mathbf{z}})$ by the approximate posterior $\mathcal{N}(\mathbf{u}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ obtained in the previous step. The expressions for the update of the natural

parameters of $q^*(\mathbf{u})$ with a new mini batch $\mathbf{y}_{\tau:\tau'}$ are

$$\boldsymbol{\eta}_1' = \boldsymbol{\eta}_1 + \sum_{t=\tau}^{\tau'} \langle \mathbf{A}_{t-1}^T \mathbf{Q}^{-1} \mathbf{x}_t \rangle_{q(\mathbf{x}_{t-1:t})}, \tag{5.29a}$$

$$\boldsymbol{\eta}_2' = \boldsymbol{\eta}_2 - \frac{1}{2} \sum_{t=\tau}^{\tau'} \langle \mathbf{A}_{t-1}^T \mathbf{Q}^{-1} \mathbf{A}_{t-1} \rangle_{q(\mathbf{x}_{t-1})}. \tag{5.29b}$$

5.7 ADDITIONAL TOPICS

5.7.1 RELATIONSHIP TO REGULARISED RECURRENT NEURAL NETWORKS

Recurrent Neural Networks (RNNs) are a class of neural networks particularly suited to modelling sequences that have seen a resurgence in recent years (see (Sutskever, 2013; LeCun et al., 2015) for recent overviews.) RNNs are dynamical systems that take a sequence as an input and produce another sequence as the output. There exist many architectures for RNNs. However, recent trends (see references above) focus on architectures that correspond to parametric state-space models such as those presented in Section 2.1. Although RNNs are not probabilistic models, they are attractive since they are computationally very efficient and have proved to learn insightful representations from sequence data.

GP-SSMs are nonparametric models which can model complex state transition functions. However, as it happens in GP regression and classification, it is often convenient to resort to approximations and limit the model capacity when dealing with large datasets. Our variational approach to learn sparse GP-SSMs is a possible approximation. By obtaining a posterior over the *finite* dimensional vector of inducing points $\mathbf{u}$, we implicitly define a posterior over the state transition function. The result is a posterior distribution over transition functions. Following a result from Section 5.7.2.1, the particular choice of variational distribution used in this chapter results in an evidence lower bound that is equivalent to having used the following state transition density

$$p(\mathbf{f}_t \mid \mathbf{x}_{t-1}, \mathbf{u}) = \mathcal{N}(\mathbf{f}_t \mid \mathbf{A}_{t-1}\mathbf{u}, \mathbf{B}_{t-1}), \tag{5.30}$$

with a prior over the inducing points

$$p(\mathbf{u}) = \mathcal{N}(\mathbf{u} \mid \mathbf{0}, \mathbf{K}_{\mathbf{z}}). \tag{5.31}$$

The mean of the predictive distribution can be expressed as

$$\mathbf{A}_{t-1}\mathbf{u} = \mathbf{K}_{\mathbf{x}_{t-1},\mathbf{z}}\mathbf{K}_{\mathbf{z}}^{-1}\mathbf{u} = \sum_{i=1}^{M} k(\mathbf{x}_{t-1}, \mathbf{z}_i)\, \mathbf{w}_i, \tag{5.32}$$

where, for simplicity, we use a vector of weights $\mathbf{w} \triangleq \mathbf{K}_{\mathbf{z}}^{-1}\mathbf{u}$.[2] From the equa-

[2]Translating the prior over inducing inputs to the weights we obtain a prior $p(\mathbf{w}) =$

tions above, we can view the mean of the transition density as a neural network (e.g. a Radial Basis Function) mapping $\mathbf{x}_{t-1}$ to $\mathbf{f}_t$ with M hidden units, where $\mathbf{w}$ is the vector of hidden to output weights and the kernel function is the hidden unit transfer function.

5.7.2 VARIATIONAL LEARNING IN RELATED MODELS

In the following, we examine the application of a similar variational evidence lower bound on models related to the GP-SSM.

5.7.2.1 MARKOVIAN MODEL WITH HETEROSCEDASTIC NOISE

Although not strictly a GP-SSM, it is also interesting to consider a model where the state transitions are independent of each other given the inducing variables

$$p(\mathbf{f}_{1:T}, \mathbf{x}_{0:T}|\mathbf{u}) = p(\mathbf{x}_0) \prod_{t=1}^{T} \mathcal{N}\big(\mathbf{f}_t|\mathbf{A}(\mathbf{x}_{t-1})\mathbf{u}, \mathbf{B}(\mathbf{x}_{t-1})\big)\, p(\mathbf{x}_t|\mathbf{f}_t). \tag{5.33}$$

This model can be interpreted as a parametric model where $\mathbf{A}(\mathbf{x}_{t-1})\mathbf{u}$ is a deterministic transition function and $\mathbf{B}(\mathbf{x}_{t-1})$ is the covariance matrix of an heteroscedastic process noise. The model is parameterised by the inducing inputs $\mathbf{z}$ and inducing points $\mathbf{u}$. Note that this process noise is *independent* between any two time steps. Maximum likelihood learning and inference in such a parametric model has been studied in detail in (Turner et al., 2010; McHutchon, 2014).

If we derive an evidence lower bound with using an analogous procedure to that presented in Section 5.2 using the variational distribution

$$q(\mathbf{x}, \mathbf{f}, \mathbf{u}) = q(\mathbf{u})q(\mathbf{x}) \prod_{t=1}^{T} \mathcal{N}\big(\mathbf{f}_t|\mathbf{A}(\mathbf{x}_{t-1})\mathbf{u}, \mathbf{B}(\mathbf{x}_{t-1})\big), \tag{5.34}$$

the lower bound becomes the *same* as the one obtained in Equation (5.4) for a GP-SSM.

5.7.2.2 BASIS FUNCTION MODEL

Another interesting nonlinear model can be created using a parametric state transition function of the form $f(\mathbf{x}) = \boldsymbol{\phi}(\mathbf{x})^\top \mathbf{w}$. Where $\mathbf{w}$ is a vector of weights and $\boldsymbol{\phi}(\mathbf{x})$ is vector of basis functions evaluated at $\mathbf{x}$. The state transition density is

$$p(\mathbf{x}_t \mid \mathbf{x}_{t-1}, \mathbf{w}) = \mathcal{N}\big(\mathbf{x}_t|\boldsymbol{\phi}(\mathbf{x}_{t-1})^\top \mathbf{w}, \mathbf{Q}\big). \tag{5.35}$$

Note that, as opposed to the model in Section 5.7.2.1, these transition dynamics are not heteroscedastic. Here, the transition from one state to the next is the consequence of a deterministic nonlinear mapping plus a Gaussian noise of constant covariance.

$\mathcal{N}(\mathbf{0}, \mathbf{K}_{\mathbf{z}}^{-1})$.

Ghahramani and Roweis (1999) used EM to find maximum likelihood solutions for this nonlinear state-model where $\boldsymbol{\phi}(\mathbf{x})$ were Gaussian radial basis functions.

It is possible to represent a GP-SSM in the form of a basis function model by approximating the Gaussian process with a truncated basis expansion, e.g. (Lázaro-Gredilla et al., 2010; Solin and Särkkä, 2014; Svensson et al., 2015). Those approximations put a Gaussian prior on the weights $p(\mathbf{w}) = \mathcal{N}(\mathbf{0}, \mathbf{S})$.

The joint distribution of a state-space model with such a prior over the transition dynamics is

$$\begin{aligned} p(\mathbf{y}, \mathbf{x}, \mathbf{w}) &= p(\mathbf{w})p(\mathbf{x}_0)\prod_{t=1}^{T} p(\mathbf{y}_t \mid \mathbf{x}_t)p(\mathbf{x}_t \mid \mathbf{x}_{t-1}, \mathbf{w}) \\ &= \mathcal{N}(\mathbf{w} \mid \mathbf{0}, \mathbf{S})\, p(\mathbf{x}_0) \prod_{t=1}^{T} p(\mathbf{y}_t \mid \mathbf{x}_t)\, \mathcal{N}\big(\mathbf{x}_t | \boldsymbol{\phi}(\mathbf{x}_{t-1})^\top \mathbf{w}, \mathbf{Q}\big). \end{aligned} \tag{5.36}$$

Using the variational distribution

$$q(\mathbf{x}, \mathbf{w}) = q(\mathbf{x})q(\mathbf{w}), \tag{5.37}$$

we can obtain the following lower bound on the evidence

$$\begin{aligned} \mathcal{L}(q(\mathbf{w}), q(\mathbf{x}), \boldsymbol{\theta}) =& \langle \log \frac{p(\mathbf{w})}{q(\mathbf{w})} \rangle_{q(\mathbf{w})} + \langle \log \frac{1}{q(\mathbf{x})} \rangle_{q(\mathbf{x})} + \langle \log p(\mathbf{x}_0) \rangle_{q(\mathbf{x}_0)} \\ &+ \langle \log \prod_{t=1}^{T} p(\mathbf{x}_t | \mathbf{x}_{t-1}, \mathbf{w}) \rangle_{q(\mathbf{x},\mathbf{w})} + \langle \log \prod_{t=1}^{T} p(\mathbf{y}_t | \mathbf{x}_t) \rangle_{q(\mathbf{x})} \\ =& -\operatorname{KL}(q(\mathbf{w}) \| p(\mathbf{w})) + \mathcal{H}(q(\mathbf{x})) + \langle \log p(\mathbf{x}_0) \rangle_{q(\mathbf{x}_0)} \\ &+ \sum_{t=1}^{T} \Big\{ \langle \mathcal{N}\big(\mathbf{x}_t | \boldsymbol{\phi}(\mathbf{x}_{t-1})^\top \mathbf{w}, \mathbf{Q}\big) \rangle_{q(\mathbf{x})q(\mathbf{w})} + \langle \log p(\mathbf{y}_t | \mathbf{x}_t) \rangle_{q(\mathbf{x})} \Big\}, \end{aligned} \tag{5.38}$$

which results in the optimal distributions

$$q^*(\mathbf{w}) \propto p(\mathbf{w}) \prod_{t=1}^{T} \exp \langle \log \mathcal{N}(\mathbf{x}_t | \boldsymbol{\phi}(\mathbf{x}_{t-1})^\top \mathbf{w}, \mathbf{Q}) \rangle_{q(\mathbf{x}_{t-1:t})}, \tag{5.39a}$$

$$q^*(\mathbf{x}) \propto p(\mathbf{x}_0) \prod_{t=1}^{T} p(\mathbf{y}_t | \mathbf{x}_t) \exp\{-\frac{1}{2} \operatorname{tr}\big(\mathbf{Q}^{-1} \boldsymbol{\phi}(\mathbf{x}_{t-1})^\top \boldsymbol{\Sigma} \boldsymbol{\phi}(\mathbf{x}_{t-1})^\top\big)\}\, \mathcal{N}(\mathbf{x}_t | \boldsymbol{\phi}(\mathbf{x}_{t-1})^\top \boldsymbol{\mu}, \mathbf{Q}), \tag{5.39b}$$

where $q^*(\mathbf{w})$ is, conveniently, a Gaussian distribution which we parameterise with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$. Note how in these expressions the matrix $\boldsymbol{\phi}(\mathbf{x}_{t-1})^\top$ takes the role of $\mathbf{A}_{t-1}$ in the sparse GP-SSM and there is no contribution equivalent to that of $\mathbf{B}_{t-1}$.

5.7.2.3 GP-SSM WITH TRANSITION AND EMISSION GPS

A state space model having a Gaussian process prior over the state transition function *and* the emission/observation function can be represented by

$$f(\mathbf{x}) \sim \mathcal{GP}\big(m_f(\mathbf{x}), k_f(\mathbf{x}, \mathbf{x}')\big), \tag{5.40a}$$
$$g(\mathbf{x}) \sim \mathcal{GP}\big(m_g(\mathbf{x}), k_g(\mathbf{x}, \mathbf{x}')\big), \tag{5.40b}$$
$$\mathbf{x}_0 \sim p(\mathbf{x}_0) \tag{5.40c}$$
$$\mathbf{x}_t \mid \mathbf{f}_t \sim \mathcal{N}(\mathbf{x}_t \mid \mathbf{f}_t, \mathbf{Q}), \tag{5.40d}$$
$$\mathbf{y}_t \mid \mathbf{g}_t \sim \mathcal{N}(\mathbf{y}_t \mid \mathbf{g}_t, \mathbf{R}), \tag{5.40e}$$

where we have used $\mathbf{f}_t \triangleq f(\mathbf{x}_{t-1})$ and $\mathbf{g}_t \triangleq g(\mathbf{x}_t)$. If the transition GP is augmented with inducing variables $\mathbf{u}$ and the emission GP is augmented with $\mathbf{v}$, we obtain the following joint distribution of the model

$$p(\mathbf{y}, \mathbf{x}, \mathbf{f}, \mathbf{u}, \mathbf{g}, \mathbf{v}) = p(\mathbf{g}|\mathbf{x}, \mathbf{v})\, p(\mathbf{x}, \mathbf{f}|\mathbf{u})\, p(\mathbf{u})\, p(\mathbf{v}) \prod_{t=1}^{T} p(\mathbf{y}_t|\mathbf{g}_t), \tag{5.41}$$

where $p(\mathbf{x}, \mathbf{f}|\mathbf{u})$ is the same as in the model presented in the paper and $p(\mathbf{g}|\mathbf{x}, \mathbf{v})$ is straightforward since it is conditioned on all the states.

We use the following variational distribution over latent variables

$$q(\mathbf{x}, \mathbf{f}, \mathbf{u}, \mathbf{g}, \mathbf{v}) = q(\mathbf{u})q(\mathbf{v})q(\mathbf{x})p(\mathbf{g}|\mathbf{x}, \mathbf{v}) \prod_{t=1}^{T} p(\mathbf{f}_t|\mathbf{f}_{1:t-1}, \mathbf{x}_{0:t-1}, \mathbf{u}). \tag{5.42}$$

Terms with latent variables inside kernel matrices cancel inside the log

$$\begin{aligned}
\log p(\mathbf{y}|\boldsymbol{\theta}) &\geq \langle \log \frac{p(\mathbf{u})p(\mathbf{v})p(\mathbf{x}_0)\prod_{t=1}^{T} p(\mathbf{y}_t|\mathbf{g}_t)p(\mathbf{x}_t|\mathbf{f}_t)}{q(\mathbf{u})q(\mathbf{v})q(\mathbf{x})} \rangle_{q(\mathbf{x},\mathbf{f},\mathbf{u},\mathbf{g},\mathbf{v})} \\
&= -\mathrm{KL}(q(\mathbf{u})\|p(\mathbf{u})) - \mathrm{KL}(q(\mathbf{v})\|p(\mathbf{v})) + \mathcal{H}(q(\mathbf{x})) + \langle \log p(\mathbf{x}_0) \rangle_{q(\mathbf{x})} \\
&+ \sum_{t=1}^{T} \Big\{ \big\langle \langle \log p(\mathbf{x}_t|\mathbf{f}_t) \rangle_{p(\mathbf{f}_t|\mathbf{x}_{t-1},\mathbf{u})} \big\rangle_{q(\mathbf{x})q(\mathbf{u})} \\
&+ \big\langle \langle \log p(\mathbf{y}_t|\mathbf{g}_t) \rangle_{p(\mathbf{g}_t|\mathbf{x}_t,\mathbf{v})} \big\rangle_{q(\mathbf{x})q(\mathbf{v})} \Big\}.
\end{aligned}$$

The optimal distribution $q^*(\mathbf{u})$ is the same as in Equation (5.14) and the optimal variational distribution of the emission inducing variables is a Gaussian distribution

$$q^*(\mathbf{v}) \propto p(\mathbf{v}) \prod_{t=1}^{T} \exp\{\langle \log \mathcal{N}(\mathbf{y}_t|\mathbf{C}_t\,\mathbf{v}, \mathbf{R}) \rangle_{q(\mathbf{x}_t)}\} \tag{5.43}$$

where

$$\begin{aligned}
\mathbf{C}_t &= \mathbf{K}_{t,\mathbf{v}}\mathbf{K}_{\mathbf{v}}^{-1}, \\
\mathbf{D}_t &= \mathbf{K}_{t,t} - \mathbf{K}_{t,\mathbf{v}}\mathbf{K}_{\mathbf{v}}^{-1}\mathbf{K}_{\mathbf{v},t}.
\end{aligned}$$

The optimal variational distribution of the state trajectory is

$$q^*(\mathbf{x}) \propto p(\mathbf{x}_0) \prod_{t=1}^{T} \exp\{-\frac{1}{2}\mathrm{tr}(\mathbf{Q}^{-1}(\mathbf{B}_{t-1} + \mathbf{A}_{t-1}\boldsymbol{\Sigma}\mathbf{A}_{t-1}{}^T)) - \frac{1}{2}\mathrm{tr}(\mathbf{R}^{-1}(\mathbf{D}_t + \mathbf{C}_t\boldsymbol{\Lambda}\mathbf{C}_t{}^T))\}$$
$$\mathcal{N}(\mathbf{x}_t|\mathbf{A}_{t-1}\boldsymbol{\mu}, \mathbf{Q})\,\mathcal{N}(\mathbf{y}_t|\mathbf{C}_t\boldsymbol{\nu}, \mathbf{R}), \tag{5.44}$$

where we have used $q(\mathbf{v}) = \mathcal{N}(\boldsymbol{\nu}, \boldsymbol{\Lambda})$.

5.7.3 ARBITRARY MEAN FUNCTION CASE

To simplify the interpretation of the equations, the rest of the chapter has been written for GP priors with a zero mean function. This section presents the expressions for the arbitrary mean function case. When a rough model of the dynamics is available it can be used as a mean function of the GP prior. This helps in obtaining meaningful distributions over the state trajectories from the first learning iterations. Recall that we use the shorthand notation $\mathbf{m}_\mathbf{x} \triangleq m_f(\mathbf{x})$.

EVIDENCE LOWER BOUND The evidence lower bound has the same expression as for the zero-mean case except for the terms:

$$p(\mathbf{u}) = \mathcal{N}(\mathbf{u} \mid \mathbf{m}_\mathbf{z}, \mathbf{K}_\mathbf{z}), \tag{5.45a}$$
$$\Phi(\mathbf{x}_{t-1:t}, \mathbf{u}) = -\frac{1}{2}\mathrm{tr}(\mathbf{Q}^{-1}\mathbf{B}_{t-1}) + \log \mathcal{N}(\mathbf{x}_t \mid \mathbf{m}_{\mathbf{x}_{t-1}} + \mathbf{A}_{t-1}\mathbf{u}, \mathbf{Q}), \tag{5.45b}$$

The ELBO becomes

$$\begin{aligned}\mathcal{L}(q(\mathbf{u}), q(\mathbf{x}), \boldsymbol{\theta}) = &- \mathrm{KL}(q(\mathbf{u})\|p(\mathbf{u})) + \mathcal{H}(q(\mathbf{x})) + \langle \log p(\mathbf{x}_0)\rangle_{q(\mathbf{x}_0)} \\ &+ \sum_{t=1}^{T} \Big\{ -\frac{1}{2}\langle \mathrm{tr}(\mathbf{Q}^{-1}(\mathbf{B}_{t-1} + \mathbf{A}_{t-1}\boldsymbol{\Sigma}\mathbf{A}_{t-1}^\top))\rangle_{q(\mathbf{x}_{t-1})} \\ &+ \langle \log \mathcal{N}(\mathbf{x}_t \mid \mathbf{m}_{\mathbf{x}_{t-1}} + \mathbf{A}_{t-1}\boldsymbol{\mu}, \mathbf{Q})\rangle_{q(\mathbf{x}_{t-1:t})} \\ &+ \langle \log p(\mathbf{y}_t \mid \mathbf{x}_t)\rangle_{q(\mathbf{x}_t)} \Big\}.\end{aligned} \tag{5.46}$$

OPTIMAL VARIATIONAL DISTRIBUTION FOR $\mathbf{u}$ The natural parameters for the (Gaussian) optimal distribution $q^*(\mathbf{u})$ are

$$\boldsymbol{\eta}_1 = \sum_{t=1}^{T} \langle (\mathbf{x}_t - \mathbf{m}_{\mathbf{x}_{t-1}})^T \mathbf{Q}^{-1}\mathbf{A}_{t-1}\rangle_{q(\mathbf{x}_{t-1:t})} + \mathbf{m}_\mathbf{z}^\top \mathbf{K}_\mathbf{z}^{-1}, \tag{5.47a}$$
$$\boldsymbol{\eta}_2 = -\frac{1}{2}\left(\mathbf{K}_\mathbf{z}^{-1} + \sum_{t=1}^{T} \langle \mathbf{A}_{t-1}^T \mathbf{Q}^{-1}\mathbf{A}_{t-1}\rangle_{q(\mathbf{x}_{t-1})}\right). \tag{5.47b}$$

OPTIMAL VARIATIONAL DISTRIBUTION FOR $\mathbf{x}$

$$q^*(\mathbf{x}) \propto p(\mathbf{x}_0) \prod_{t=1}^{T} p(\mathbf{y}_t|\mathbf{x}_t) \exp\{-\frac{1}{2}\mathrm{tr}\big(\mathbf{Q}^{-1}(\mathbf{B}_{t-1} + \mathbf{A}_{t-1}\boldsymbol{\Sigma}\mathbf{A}_{t-1}^T)\big)\}\,\mathcal{N}(\mathbf{x}_t|\mathbf{m}_{\mathbf{x}_{t-1}} + \mathbf{A}_{t-1}\boldsymbol{\mu}, \mathbf{Q}). \tag{5.48}$$

PREDICTIVE DISTRIBUTION

$$p(\mathbf{f}_*|\mathbf{x}_*, \mathbf{y}) \approx \mathcal{N}(\mathbf{f}_*|\mathbf{m}_{\mathbf{x}_*} + \mathbf{A}_*\boldsymbol{\mu}, \mathbf{B}_* + \mathbf{A}_*\boldsymbol{\Sigma}\mathbf{A}_*^\top). \tag{5.49}$$

5.8 EXPERIMENTS

This section showcases the ability of variational GP-SSMs to perform approximate Bayesian learning of nonlinear dynamical systems. In particular, we aim to demonstrate: 1) the ability to learn the inherent nonlinear dynamics of a system, 2) the application in cases where the latent states have higher dimensionality than the observations, and 3) the use of non-Gaussian likelihoods.

5.8.1 1D NONLINEAR SYSTEM

We apply our variational learning procedure presented above to the one-dimensional nonlinear system described by $p(x_{t+1}|x_t) = \mathcal{N}(f(x_t), 1)$ and $p(y_t|x_t) = \mathcal{N}(x_t, 1)$ where the transition function is $x_t + 1$ if $x < 4$ and $-4x_t + 21$ if $x \geq 4$. Its pronounced kink makes it challenging to learn. Our goal is to find a posterior distribution over this function using a GP-SSM with Matérn covariance function. To solve the expectations with respect to the approximate smoothing distribution $q(\mathbf{x})$ we use a bootstrap particle fixed-lag smoother with 1000 particles and a lag of 10.

In Table 5.1, we compare our method (Variational GP-SSM) against the PMCMC sampling procedure from (Frigola et al., 2013) taking 100 samples and 10 burn in samples. As in (Frigola et al., 2013), the sampling exhibited very good mixing with 20 particles. We also compare to an auto-regressive model based on Gaussian process regression (Quiñonero-Candela et al., 2003) of order 5 with Matérn ARD covariance function with and without FITC approximation. Finally, we use a linear subspace identification method (N4SID, (Overschee and Moor, 1996)) as a baseline for comparison. The PMCMC training offers the best test performance from all methods using 500 training points at the cost of substantial train and test time. However, if more data is available ($T = 10^4$) the stochastic variational inference procedure can be very attractive since it improves test performance while having a test time that is independent of the training set size. The reported SVI performance has been obtained with mini-batches of 100 time-steps.

5.8.2 NEURAL SPIKE TRAIN RECORDINGS

We now turn to the use of SSMs to learn a simple model of neural activity in rats' hippocampus. We use data in neuron cluster 1 (the most active) from experiment ec013.717 in (Mizuseki et al., 2013). In some regions of the time series, the action potential spikes show a clear pattern where periods of rapid spiking are followed by periods of very little spiking. We wish to model this behaviour as an autonomous nonlinear dynamical system (i.e. one not driven by exter-

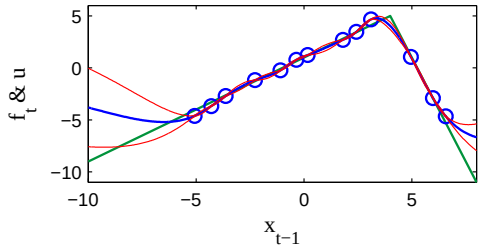


Figure 5.1: Posterior distribution over latent state transition function (green: ground truth, blue: posterior mean, red: mean ± 1 standard deviation).

	Test RMSE	$\log p(\mathbf{x}_{t+1}^{\text{test}} \mid \mathbf{x}_t^{\text{test}}, \mathbf{y}_{0:T}^{\text{tr}})$	Train time	Test time
Variational GP-SSM	1.15	-1.61	2.14 min	0.14 s
Var. GP-SSM (SVI, $T = 10^4$)	1.07	-1.47	4.12 min	0.14 s
PMCMC GP-SSM	1.12	-1.57	547 min*	421 s
GP-NARX	1.46	-1.90	0.22 min	3.85 s
GP-NARX + FITC	1.47	-1.90	0.17 min	0.23 s
Linear (N4SID)	2.35	-2.30	0.01 min	0.11 s

*Table 5.1: Experimental evaluation of 1D nonlinear system. Unless otherwise stated, training times are reported for a dataset with $T = 500$ and test times are given for a test set with 10^5 data points. All pre-computations independent on test data are performed before timing the "test time". Predictive log likelihoods are the average over the full test set. * The PMCMC code used for these experiments did not use fast updates-downdates of the Cholesky factors during training (Section 4.4.2). This does* not *affect test times.*

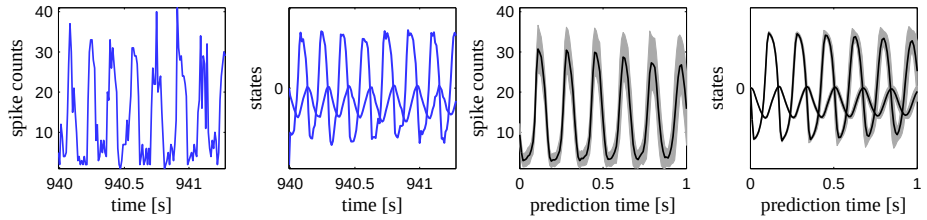


Figure 5.2: From left to right: 1) part of the observed spike count data, 2) sample from the corresponding smoothing distribution, 3) predictive distribution of spike counts obtained by simulating the posterior dynamical from an initial state, and 4) corresponding latent states.

nal inputs). Many parametric models of nonlinear neuron dynamics have been proposed (Izhikevich, 2000) but our goal here is to learn a model from data without using any biological insight. We use a GP-SSM with a structure such that it is the discrete-time analog of a second order nonlinear ordinary differential equation: two states one of which is the derivative of the other. The observations are spike counts in temporal bins of 0.01 second width. We use a Poisson likelihood relating the spike counts to the second latent state

$$y_t | \mathbf{x}_t \sim \text{Poisson}(\exp(\alpha \mathbf{x}_t^{(2)} + \beta)).$$

We find a posterior distribution for the state transition function using our variational GP-SSM approach. Smoothing is done with a fixed-lag particle smoother and training until convergence takes approximately 50 iterations of Algorithm 3. Figure 5.2 shows a part of the raw data together with an approxi-

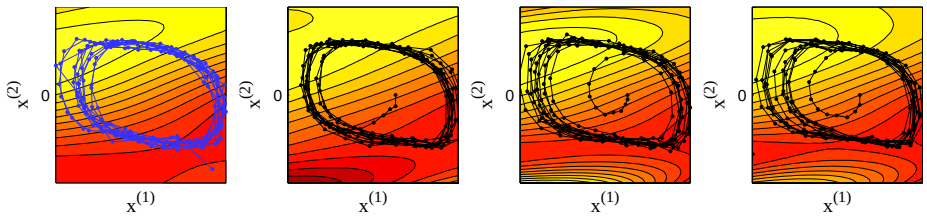


Figure 5.3: Contour plots of the state transition function $\mathbf{x}_{t+1}^{(2)} = f(\mathbf{x}_t^{(1)}, \mathbf{x}_t^{(2)})$, and trajectories in state space. Left: mean posterior function and trajectory from smoothing distribution. Other three panels: transition functions sampled from the posterior and trajectories simulated conditioned on the corresponding sample. Those simulated trajectories start inside the limit cycle and are naturally attracted towards it. Note how function samples are very similar in the region of the limit cycle.

mate sample from the smoothing distribution during the same time interval. In addition, we show the distribution over predictions made by chaining 1-step-ahead predictions. To make those predictions we have switched off process noise ($\mathbf{Q} = \mathbf{0}$) to show more clearly the effect of uncertainty in the state transition function. Note how the frequency of roughly 6 Hz present in the data is well captured. Figure 5.3 shows how the limit cycle corresponding to a nonlinear dynamical system has been captured (see caption for details).

Chapter 6

Filtered Auto-Regressive Gaussian Process Models

This chapter presents a novel approach to learning nonlinear models of time series which integrates an automatic data preprocessing step with the training of a nonlinear auto-regressive model (Frigola and Rasmussen, 2013). The resulting algorithm is extremely simple to implement and can quickly learn a posterior over nonlinear dynamical systems from long time series data. This posterior can be used for prediction, extrapolation and other related tasks. For instance, one could learn a model of a nonlinear dynamical system from input/output data and use this model to design a feedback controller to stabilise the system.

6.1 Introduction

Gaussian Process State-Space Models are very general and flexible models of nonlinear dynamical systems. One of their main strengths is their ability to model dynamics in an abstract state space that is different to the observation space. However, this poses the challenge of needing to infer the posterior distribution over the state trajectories given the observed time series, i.e. the *smoothing* distribution. This operation can be performed very efficiently with a Kalman smoother when the state-space model is linear and Gaussian. However, for general nonlinear and/or non-Gaussian models, one needs to resort to approximations such as the Extended Kalman Smoother, the Particle Smoother or Particle MCMC. Depending on the particular characteristics of the problem at hand, a given method can perform brilliantly or fail miserably.

In this section we present a method based on an auto-regressive nonlinear model (Section 2.3.6) where the difficult task of state smoothing is sidestepped. This leads to learning with a simple algorithm, less computational cost and more robustness due to avoiding the use of a state smoother.

However, auto-regressive models are less general than state-space models. They are not able to separate between process noise (e.g. turbulence changing

the actual trajectory of an aircraft) from observation noise (e.g. random errors in the measurement of aircraft position.) In auto-regressive models, all noise can, in principle, modify the future trajectory of the system. Whereas in state-space models the state trajectory is independent of observation noise.

6.1.1 END-TO-END MACHINE LEARNING

The algorithm presented in this chapter recognises the need to take into account observation noise. However, we use an auto-regressive model that has no concept of observation noise. The conventional approach in system identification is to manually *preprocess* the data to remove observation noise before learning the model (Ljung, 1999). This can be attempted, for instance, by applying a low-pass filter on time-domain signals to remove high-frequency noise. Instead, our approach is to *automatically* tune a parameterised preprocessing step *simultaneously* with the learning of the nonlinear auto-regressive model. For instance, if we choose to preprocess the data with a low-pass filter, the bandwidth of the filter can be tuned while learning the model.

According to Blocker and Meng (2013), "decisions made in preprocessing constrain all later analyses and are typically irreversible". Our approach is to integrate preprocessing with model learning in order to perform the right amount of preprocessing to obtain good predictive performance.

6.1.2 ALGORITHMIC WEAKENING

Our approach can also be interpreted within an *algorithmic weakening* framework (Jordan, 2013). Algorithmic weakening refers to having a hierarchy of models with different computational complexities and, as data accrue, a simpler model is used. Jordan (2013) states that for a given quality of inference "it may be possible to save on computation because of the growth of statistical power as problem instances grow in size." In other words, the fact that more data are available compensates for the use of simpler models and leads to less computation for the same inferential quality.

Although state-space models generalise auto-regressive models, they are generally harder to learn due to the need to solve a filtering or smoothing problem as an intermediate step. Auto-regressive models where data has been preprocessed to remove as much observation noise as possible can be seen as a step down from a hierarchy of models: they are not as general but they are typically faster to train. As a consequence, for large datasets, they can be faster to reach a certain level of inferential quality.

Jordan (2013) does not touch the issues of ease of use and robustness of algorithms. These are also important practical factors to take into account. Unfortunately, algorithms for nonlinear filtering and smoothing need to be tuned to each particular application if computational efficiency is important. The fact that no state inference is necessary in auto-regressive models leads to simpler algorithms that can be more easily used by non-experts.

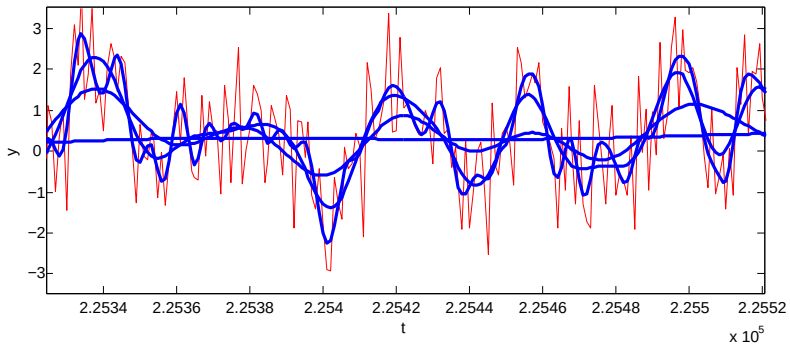


Figure 6.1: An original signal (red) filtered with a low-pass filter using different settings for the cut-off frequency (blue). The filter is a zero-phase fourth order Butterworth low-pass filter.

6.2 THE GP-FNARX MODEL

In this section, we describe the GP-FNARX model for nonlinear system identification based on a nonlinear auto-regressive exogenous (NARX) model with filtered regressors (F) where the nonlinear regression problem is tackled using sparse Gaussian processes (GP). The approach integrates preprocessing with model learning into a fully automated procedure that yields a posterior over dynamical systems. Training is performed by simultaneously maximising the *marginal likelihood* of the probabilistic regression model with respect to the preprocessing parameters and the GP hyper-parameters.

Auto-regressive models with exogenous inputs attempt to predict the present output by considering it a function of past inputs and outputs:

$$\mathbf{y}_t = f(\mathbf{y}_{t-1}, \mathbf{y}_{t-2}, ..., \mathbf{u}_{t-1}, ...) + \boldsymbol{\delta}_t. \tag{6.1}$$

The identification problem is then posed as a regression task. In other words, we want to infer the function $\mathbf{y} = f(\mathbf{x})$ from a finite number of examples $\{\mathbf{y}_i, \mathbf{x}_i\}$, where $\mathbf{x}_i = (\mathbf{y}_{i-1}, \mathbf{y}_{i-2}, ..., \mathbf{u}_{i-1}, ...)$. In our approach, we place a GP prior on the function $f(\mathbf{x})$ and use sparse GP regression techniques to learn it from observed data.

If the input and output signals to the dynamical system are noisy, we face an *errors-in-variables* problem since the regressors $\mathbf{x}$ are noisy. Noise in the regressors makes the regression problem particularly hard (McHutchon and Rasmussen, 2011; Damianou and Lawrence, 2015). This is one of the reasons why, in practical system identification applications, input signals are typically preprocessed before learning a model of the system dynamics. For instance, one can carefully low-pass filter the signals to remove high-frequency noise. In Fig. 6.1, we show the results of filtering an originally noisy signal with a zero-phase fourth order Butterworth low-pass filter. A signal with $T = 10^6$ can be filtered in only 0.05 seconds using Matlab on a mid-range desktop. This is orders of magnitude faster than approximate smoothing in a nonlinear state-space model.

We will consider *any* data preprocessing function applied to the input and

output signals

$$(\hat{\mathbf{y}}, \hat{\mathbf{u}}) = h(\mathbf{y}, \mathbf{u}, \boldsymbol{\omega}) \tag{6.2}$$

where the preprocessed signals vary *smoothly* with respect to a vector of preprocessing parameters $\boldsymbol{\omega}$. This smoothness condition is imposed in order to obtain a probabilistic model with a differentiable marginal likelihood with respect to $\boldsymbol{\omega}$.

We can rephrase the auto-regressive model in terms of the preprocessed regressors:

$$\mathbf{y}_t = f(\hat{\mathbf{y}}_{t-1}, \hat{\mathbf{y}}_{t-2}, ..., \hat{\mathbf{u}}_{t-1}, ...). \tag{6.3}$$

Note that the left hand side term is *not* preprocessed.

6.2.1 CHOICE OF PREPROCESSING AND COVARIANCE FUNCTIONS

6.2.1.1 CAUSAL PREPROCESSING

If trained GP-FNARX models are intended to make online predictions, it is convenient to use causal preprocessing. Causal preprocessing consists in the use of a preprocessing algorithm whose output depends on past and current inputs but not on future inputs. For example, a conventional moving average filter is a causal filter but the zero-phase filter in Figure 6.1 is non-causal.

Using a causal preprocessing stage mimics the kind of filtering that can be realistically performed online. Therefore, the preprocessing parameters tuned during training can be used on exactly the same filtering algorithm when the model is deployed.

6.2.1.2 COVARIANCE FUNCTION

An alternative to causal preprocessing on the full input and output signals is to embed a filter in the covariance function of the Gaussian process. For example, a kernel could incorporate a moving average filter

$$k(\mathbf{y}_{a:b}, \mathbf{y}'_{a:b}) = \tilde{k}(\sum_{i=a}^{b} y_i, \sum_{i=a}^{b} y'_i) \tag{6.4}$$

or more sophisticated parameterised preprocessing

$$k(\mathbf{y}_{a:b}, \mathbf{y}'_{a:b}) = \tilde{k}\big(h(\mathbf{y}_{a:b}, \boldsymbol{\omega}), h(\mathbf{y}'_{a:b}, \boldsymbol{\omega})\big). \tag{6.5}$$

In such kernels, preprocessing can be interpreted as an explicit intermediate featurisation of the GP index set. In fact, $h(\mathbf{y}_{a:b}, \boldsymbol{\omega})$ does not need to have the same dimensionality as $\mathbf{y}_{a:b}$. For example, it could be a filtered version of $\mathbf{y}_{a:b}$ sub-sampled at a few locations.

This formalisation of preprocessing converts the problem of tuning preprocessing parameters into a kernel search problem. (See (Duvenaud, 2014) for an excellent tutorial on expressing structure with kernels and an overview of

kernel search.) In our Gaussian process framework, kernel search can be performed via maximisation of the marginal likelihood.

Note that, within a kernel, it is feasible to relax the requirement for a causal filter even when predictions will be made online. For instance, consider a linear digital filter

$$k(\mathbf{y}_{a:b}, \mathbf{y}'_{a:b}) = \tilde{k}\big(\mathbf{H}\mathbf{y}_{a:b}, \mathbf{H}\mathbf{y}'_{a:b}\big), \tag{6.6}$$

where $\mathbf{H}$ is the linear digital filter matrix. A filtering and subsampling strategy could be implemented using a wide and short matrix. There is no need for this filter to be causal within the $a : b$ range, therefore $\mathbf{H}$ does not require a triangular structure. $\mathbf{H}$ could be relatively large, e.g. 4×100, but parameterised very lightly, e.g. with only a scalar bandwidth parameter. It would also be possible to start training with such a light parameterisation and once a rough bandwidth has been found switch to optimisation of all the elements in $\mathbf{H}$. This would let the algorithm "discover" the best combination of filter type, subsampling strategy and hyper-parameters for the base kernel $\tilde{k}(\cdot, \cdot)$. A similar filtering strategy could be implemented for the mean function of the GP.

6.3 OPTIMISATION OF THE MARGINAL LIKELIHOOD

In Section 2.2.1 we described how the marginal likelihood provides a very powerful metric to perform model selection due to its ability to automatically trade off model fit and model complexity in a principled manner. Our goal here will be to maximise the marginal likelihood of the Gaussian process regression model with respect to the signal preprocessing parameters and also, simultaneously, with respect to the hyper-parameters of the GP. For convenience, we introduce $\boldsymbol{\psi} \triangleq \{\boldsymbol{\omega}, \boldsymbol{\theta}\}$ grouping the two kinds of parameters.

We will employ hill-climbing optimisation to maximise the marginal likelihood (or, equivalently, its logarithm). For notational simplicity, we group all the preprocessed regressors into a matrix $\hat{\mathbf{X}} = \hat{\mathbf{X}}(\mathbf{y}, \mathbf{u}, \boldsymbol{\omega})$. The log marginal likelihood becomes

$$\log p(\mathbf{y}|\hat{\mathbf{X}}(\boldsymbol{\omega}), \boldsymbol{\theta}). \tag{6.7}$$

Its derivatives with respect to the GP hyper-parameters

$$\frac{\partial}{\partial \boldsymbol{\theta}_j} \log p(\mathbf{y}|\hat{\mathbf{X}}(\boldsymbol{\omega}), \boldsymbol{\theta}), \tag{6.8}$$

are straightforward since we typically choose differentiable covariance functions. However, the derivatives with respect to any of the preprocessing parameters

$$\frac{\partial}{\partial \boldsymbol{\omega}_k} \log p(\mathbf{y}|\hat{\mathbf{X}}(\boldsymbol{\omega}), \boldsymbol{\theta}) \tag{6.9}$$

can be more difficult to compute since derivatives with respect to the kernel matrix also need to be computed. We can write the derivative of a single ele-

Algorithm 4 High-level pseudo-code for the GP-FNARX algorithm.

Inputs: $\mathcal{I}$ = {output signals $\mathbf{y}_{1:T}$, input signals $\mathbf{u}_{1:T}$, model order η}

1. $\boldsymbol{\psi}_0 \leftarrow \text{INITIALGUESS}(\mathcal{I})$
2. $\boldsymbol{\psi}_{\text{Opt}} \leftarrow \text{MAXIMISEMARGLIKELIHOOD}(\boldsymbol{\psi}_0, \mathcal{I})$
3. Predictive model $\leftarrow \text{PRECOMPUTEPREDICTOR}(\psi_{\text{Opt}}, \mathcal{I})$

ment of the covariance matrix as

$$\frac{\partial \mathbf{K}_{ij}}{\partial \boldsymbol{\omega}_k} = \frac{\partial k(\hat{\mathbf{x}}_i, \hat{\mathbf{x}}_j)}{\partial \boldsymbol{\omega}_k} = \frac{\partial k}{\partial \hat{\mathbf{x}}_i}\frac{\partial \hat{\mathbf{x}}_i}{\partial \boldsymbol{\omega}_k} + \frac{\partial k}{\partial \hat{\mathbf{x}}_j}\frac{\partial \hat{\mathbf{x}}_j}{\partial \boldsymbol{\omega}_k} \tag{6.10}$$

where the derivatives with respect to the regressors are straightforward to compute when using smooth covariance functions. However, the derivatives of the regressors with respect to the preprocessing parameters may be hard to compute. In any case, if the preprocessing function is smooth, the derivatives $\frac{\partial \hat{\mathbf{x}}}{\partial \boldsymbol{\omega}_k}$ can be approximated numerically by finite differences at the cost of one extra evaluation of the preprocessing function per dimension of $\boldsymbol{\omega}$.

6.4 SPARSE GPS FOR COMPUTATIONAL SPEED

Computing the marginal likelihood for datasets with more than a few thousand points becomes computationally expensive. In such settings we can use sparse Gaussian processes for regression such as FITC (Snelson and Ghahramani, 2006) or variational sparse GPs (Titsias, 2009). Their computational complexity during training is $\mathcal{O}(M^2T)$ instead of the $\mathcal{O}(T^3)$ of the full GP. After all possible pre-computations, the computational complexity of making predictions with sparse GPs is $\mathcal{O}(M)$ for the mean of the predictive distribution and $\mathcal{O}(M^2)$ for its variance. The computational efficiency comes from having *condensed* the original dataset with roughly T points into a smaller set of M inducing points.

6.5 ALGORITHM

In Algorithm 4 we present a high-level overview of the GP-FNARX algorithm. The first step consists in providing an initial guess for the unknown hyper-parameters. We have found that, in the absence of any problem-specific knowledge that could guide the initial guess, a successful data-based heuristic is to run a few steps of optimisation of the marginal likelihood with respect to the GP hyper-parameters followed by a few steps of optimisation with respect to the preprocessing parameters. By running these two steps, the algorithm can rapidly hone in the right order of magnitude for the unknown parameters.

The second step consists in a straightforward joint optimisation of the GP hyper-parameters and the preprocessing parameters. This step can be per-

formed by either using a subset of the data with a conventional GP or, preferably, on the full dataset using a sparse GP (Quiñonero-Candela and Rasmussen, 2005; Titsias, 2009).

In the third and final step, any $\mathcal{O}(T)$ operation in the sparse GP predictor can be pre-computed in order to obtain a model that is able to provide predictive means in $\mathcal{O}(M)$ and predictive variances in $\mathcal{O}(M^2)$.

The choice of the order of the auto-regressive model is not critical for the performance of the algorithm provided that two conditions are met: *i)* the order is chosen to be higher than the optimal order and *ii)* the automatic relevance determination (ARD) covariance function is chosen for the GP. This is due to the fact that the Bayesian Occam's razor embodied by the marginal likelihood is able to automatically disable regressors that are irrelevant to the task at hand. In our experiments we verified that adding hundreds of regressors on a problem of low order did not cause any overfitting and only represented a computation time penalty.

6.6 EXPERIMENTS

In this section we present an experimental evaluation of the proposed system identification method. We have used data from two nonlinear system identification benchmarks based on electric circuits: *i)* the Silverbox benchmark originating from the 2004 IFAC Symposium on Nonlinear Control Systems (NOLCOS) and *ii)* the SYSID09 Wiener-Hammerstein system identification benchmark (Schoukens et al., 2009) which has been the object of a special section in the November 2012 issue of the "Control Engineering Practice" journal (Hjalmarsson et al., 2012).

Both datasets have $>10^5$ data points and are corrupted by a very small amount of noise. For instance, the authors of the Wiener-Hammerstein dataset estimate its signal to noise ratio to be of 70 dB. Since we are attempting to demonstrate the ability of our method to cope with measurement noise, we will inject different amounts of synthetic i.i.d. Gaussian noise to the output signals to make the identification task more challenging. Being able to have original signals with little noise will be convenient to test the quality of the resulting models.

We have compared GP-FNARX against other off-the-shelf alternatives. In particular, we use several NARX models available in the Matlab System Identification toolbox (Ljung, 2012).

Following this spirit, we have avoided any tuning of our method to the particular benchmarks presented in this section. For instance, using knowledge about the underlying system of the Silverbox benchmark, we obtained better performance by adding cubic regressors into our NARX model. However, we have not used those custom regressors when reporting the performance of our model.

Regarding the preprocessing step, we have chosen a simple zero-phase sec-

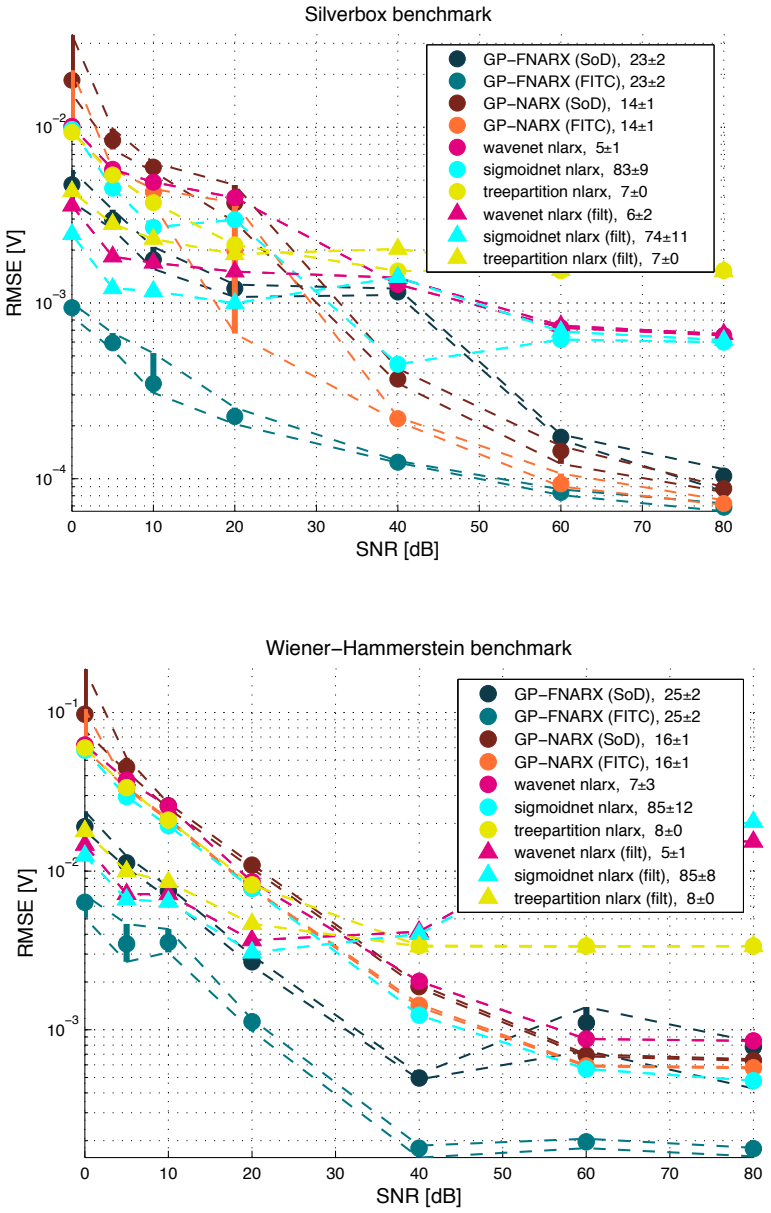


Figure 6.2: Root mean squared prediction error on the test sets as a function of the signal to noise ratio in the output signal. The mean computation time and standard deviation for each method are displayed in the legend (in seconds). Experiments are repeated 10 times, the marker is positioned at the median value and error-bars indicate the 10-90 percentile interval.

ond order Butterworth low-pass filter. In this case, the filter parameter ω represents the cut-off frequency of the filter.

In Figure 6.2 we have plotted the prediction errors on both benchmarks for a number of different models and signal-to-(synthetic)-noise ratios. We have tested the following models:

- GP-FNARX (SoD): the model presented in this paper using a subset of 512 randomly chosen points (subset of data approximation, SoD (Rasmussen and Williams, 2006)).
- GP-FNARX (FITC): the model presented in this paper using a FITC sparse GP (Quiñonero-Candela and Rasmussen, 2005) with $M = 512$ inducing points chosen at random.
- GP-NARX (SoD): same as GP-FNARX (SoD) but with no pre-filtering of the signals.
- GP-NARX (FITC): same as GP-FNARX (FITC) but with no pre-filtering of the signals.
- * nlarx: 3 different NARX models implemented in the Matlab System Identification toolbox (Ljung, 2012). Default options. No pre-filtering of the signals.
- * nlarx (filt): same as * nlarx but using the pre-filtering parameters taken from GP-FNARX (FITC) (the computation time for computing those parameters is not taken included in the reported figure).

All models have order 10 for the inputs and the outputs.

We observe that the GP-FNARX method with a FITC sparse GP provides the lowest error for all noise levels in both benchmarks. Overall, these results allow us to be optimistic with regards to the prediction capabilities of GP-FNARX models and validate the use of the marginal likelihood as a criterion for model selection in the context of automated preprocessing of the data.

The legend of Figure 6.2 reports the computation times of the experiments when performed on a machine with a 2008 Intel Core i7-920 processor at 2.67 GHz. Although the training time of the GP-FNARX model is higher than the wavenet and treepartition models, it is significantly faster than sigmoidnet yet it provides a lower prediction error.

GP models have a one degree of freedom knob which can be used to trade off accuracy with speed: the number of data points used for SoD or the number of inducing points in a sparse GP method such as FITC.

Chapter 7

Conclusions

7.1 Contributions

In this thesis, we have presented a number of contributions towards practical Bayesian learning of nonlinear time series models. Solutions have been offered to attempt to achieve a number of often conflicting goals:

Bayesian learning: All the work presented in this thesis has aimed at obtaining posteriors over nonlinear dynamical systems. These posteriors model the uncertainty about what the actual dynamical systems are given the particular time series used for training. In some settings, quantifying predictive uncertainty is useful to improve decisions made with the model. For example, our methods can be used as one of the building blocks within Bayesian reinforcement learning or for the design of adaptive-robust feedback controllers.

Nonparametric modelling: We presented learning algorithms for models of time series based on Gaussian processes. The sampling methods introduced in Chapter 4 and the GP-FNARX auto-regressive model in Chapter 6 are able to learn fully nonparametric models of time series which combine very large model capacity while avoiding the risk of overfitting.

Fast training: Throughout the thesis we have provided sparse versions for all models. Models based on sparse Gaussian processes inherit most of the properties of their fully nonparametric counterparts whilst reducing the computational complexity. In particular, they offer a convenient knob to trade off model capacity and computation time. We also presented an orthogonal approach to reduce training time for large datasets based on mini-batches (Chapter 5). Learning from mini-batches avoids having to process the full dataset at each iteration which can result in much faster optimisation when the time series are long.

Fast prediction: Monte Carlo methods from Chapter 4 can capture the full nonparametric expressivity of GP-SSMs but are slow at test time. The variational learning approach in Chapter 5 was designed to avoid this problem by approximating the posterior over nonlinear dynamical sys-

tems with a representation whose size is independent on the length of the time series.

Ease of use: The filtered nonlinear auto-regressive algorithm introduced in Chapter 6 provides a simple, robust and fast learning method that makes it well suited to application by non-experts. Its main advantage is that it avoids the expensive (and potentially difficult to tune) smoothing step that is a key part of learning nonlinear state-space models.

Most of the work in this thesis has revolved around Gaussian Process State-Space Models. We started by providing a literature review that unifies previous work related to the model. We then derived a new formulation for the model that enables straightforward sampling from the prior. The insights offered by the new formulation have led us to the creation of new Bayesian learning algorithms for the GP-SSM based on Particle MCMC and variational inference.

7.2 FUTURE WORK

The models presented in this dissertation are probabilistic in nature. They represent an idealisation of the complex system that generated the data. By using inverse probability it is possible to go from observed data to a posterior over the unobserved variables of interest. This posterior captures uncertainty which can be propagated to future predictions in a principled manner.

However, many recent successes in sequence modelling have stemmed from models that are essentially non-probabilistic[1] (Graves et al., 2013; Sutskever, 2013; LeCun et al., 2015). Although we have advocated for the use of generative probabilistic models, we should ask ourselves: what are the characteristics of non-probabilistic models such as Recurrent Neural Networks that allow them to learn such insightful representations of sequences? A tentative answer could be that their success is based on the combination of large datasets and very well engineered training algorithms.

Modelling uncertainty in probabilistic learning algorithms imposes an extra computational burden that their non-probabilistic counterparts do not have to bear. An interesting avenue of future work consists in taking the insights and techniques that have led to successful non-probabilistic methods and to apply them in the context of learning probabilistic generative models. A relatively straightforward way to achieve this is by first training computationally efficient but non-probabilistic models of time series and subsequently use them as an initialisation for probabilistic learning algorithms such as the variational GP-SSM in Chapter 5. An interesting alternative to this approach consists in reinterpreting successful algorithms for non-probabilistic models in ways that allow them to capture model uncertainty, see for instance (Gal and Ghahramani, 2015). This would capitalise on the heavy engineering already done for some non-probabilistic models and allow us to move towards the ultimate goal of providing uncertainty estimates for any prediction.

[1] In the sense that they do not model uncertainty in the parameters and states of the model.

Appendix A

Approximate Bayesian Inference

In this appendix we provide an overview of the two main learning algorithms that are used in this thesis to learn Gaussian Process State-Space Models.

A.1 Particle Markov Chain Monte Carlo

Markov Chain Monte Carlo (MCMC) methods are popular in Bayesian learning because they allow approximate learning in intractable models. They operate by simulating a Markov chain which admits the target distribution as its stationary distribution. If the target distribution is the posterior, samples obtained via MCMC can be treated as samples from the posterior distribution (with some caveats, see for example (Bishop, 2006)).

Particle MCMC (Andrieu et al., 2010) is a recent variant of MCMC that has proven to be particularly suited to sample from high-dimensional distributions with high correlations. From the various Particle MCMC algorithms introduced in (Andrieu et al., 2010), we use the Particle Gibbs algorithm. Particle Gibbs can be considered a Gibbs sampler that relies on a modified particle filter which is conditioned on a reference trajectory.

A.1.1 Particle Gibbs with Ancestor Sampling

We now introduce the detailed algorithm for Particle Gibbs with Ancestor Sampling (PGAS, Algorithm 5, Lindsten et al. (2014)). PGAS builds on Particle Gibbs by adding an additional sampling step that mitigates the effects of path degeneracy. This improves mixing and allows the use of a small number of particles (Lindsten et al., 2014).

To sample the state trajectory, PGAS makes use of a Sequential Monte Carlo-like procedure referred to as the Conditional Particle Filter with Ancestor Sampling (CPF-AS, Algorithm 6). This approach is particularly well suited to non-Markovian latent variable models, as it relies only on a forward recursion

Algorithm 5 Particle Gibbs with ancestor sampling (PGAS)

Goal: obtain samples from $p(\boldsymbol{\theta}, \mathbf{x}_{0:T} \mid \mathbf{y}_{1:T})$

1. Set $\boldsymbol{\theta}[0]$ and $\mathbf{x}_{0:T}[0]$ arbitrarily.
2. **For** $\ell \geq 1$ **do**
   (a) Draw $\boldsymbol{\theta}[\ell]$ conditionally on $\mathbf{x}_{0:T}[\ell-1]$ and $\mathbf{y}_{1:T}$.
   (b) Run CPF-AS (see Algorithm 6) targeting $p(\mathbf{x}_{0:T} \mid \boldsymbol{\theta}[\ell], \mathbf{y}_{1:T})$, conditionally on $\mathbf{x}_{0:T}[\ell-1]$.
   (c) Sample k with $\mathrm{P}(k=i) = w_T^i$ and set $\mathbf{x}_{0:T}[\ell] = \mathbf{x}_{0:T}^k$.
3. **end**

(Lindsten et al., 2014). The difference between a standard particle filter (PF) and the CPF-AS is that, for the latter, one particle at each time step is specified *a priori*. These particles will be named *reference trajectory* and be denoted by $\widetilde{\mathbf{x}}_{0:T} \triangleq \{\widetilde{\mathbf{x}}_0, ..., \widetilde{\mathbf{x}}_T\}$. We then sample according to $\mathbf{x}_t^i \sim p(\mathbf{x}_t \mid \boldsymbol{\theta}, \mathbf{x}_{0:t-1}^{\mathbf{a}_t^i})$ only for the particles $i = 1, ..., N-1$. Whereas the N-th particle is set deterministically: $\mathbf{x}_t^N = \widetilde{\mathbf{x}}_t$.

To be able to construct the N-th particle trajectory, $\mathbf{x}_t^N$ has to be associated with an ancestor particle at time $t-1$. This is done by sampling a value for the corresponding ancestor index $\mathbf{a}_t^N$. Following (Lindsten et al., 2014), the ancestor sampling probabilities are computed as

$$\widetilde{\mathbf{w}}_{t-1|T}^i \propto \mathbf{w}_{t-1}^i \frac{p(\{\mathbf{x}_{0:t-1}^i, \widetilde{\mathbf{x}}_{t:T}\}, \mathbf{y}_{1:T})}{p(\mathbf{x}_{0:t-1}^i, \mathbf{y}_{1:t-1})} \tag{A.1a}$$

$$\propto \mathbf{w}_{t-1}^i \frac{p(\{\mathbf{x}_{0:t-1}^i, \widetilde{\mathbf{x}}_{t:T}\})}{p(\mathbf{x}_{0:t-1}^i)} \tag{A.1b}$$

$$= \mathbf{w}_{t-1}^i \, p(\widetilde{\mathbf{x}}_{t:T} \mid \mathbf{x}_{0:t-1}^i). \tag{A.1c}$$

where the ratio is between the unnormalised target densities up to time T and up to time $t-1$, respectively. The second proportionality follows from the mutual conditional independence of the observations, given the states. Here, $\{\mathbf{x}_{0:t-1}^i, \widetilde{\mathbf{x}}_{t:T}\}$ refers to a path formed by concatenating the two partial trajectories. The ancestor sampling weights $\{\widetilde{\mathbf{w}}_{t-1|T}^i\}_{i=1}^N$ are then normalised to sum to 1 and the ancestor index $\mathbf{a}_t^N$ is sampled with $P(\mathbf{a}_t^N = j) = \mathbf{w}_{t-1|t}^j$.

The conditioning on a prespecified trajectory results in an important invariance property of the CPF-AS. Given $\widetilde{\mathbf{x}}_{0:T}$, let $\widetilde{\mathbf{x}}'_{0:T}$ be generated as follows:

1. Run CPF-AS from time $t=0$ to time $t=T$, conditionally on $\widetilde{\mathbf{x}}_{0:T}$.
2. Set $\widetilde{\mathbf{x}}'_{0:T}$ to one of the resulting particle trajectories according to $P(\widetilde{\mathbf{x}}'_{0:T} = \mathbf{x}_{0:T}^i) = \mathbf{w}_T^i$.

For any number of particles $N \geq 2$, this procedure defines an ergodic Markov kernel $M_{\boldsymbol{\theta}}^N(\widetilde{\mathbf{x}}'_{0:T} \mid \widetilde{\mathbf{x}}_{0:T})$, leaving the *exact* smoothing distribution $p(\mathbf{x}_{0:T} \mid \boldsymbol{\theta}, \mathbf{y}_{1:T})$ invariant (Lindsten et al., 2014). Note that this invariance holds for any $N \geq 2$, i.e. the number of particles that are used only affects the mixing rate of the ker-

Algorithm 6 CPF-AS, conditioned on $\widetilde{\mathbf{x}}_{0:T}$

1. **Initialise:**
   (a) Draw $\mathbf{x}_0^i \sim p(\mathbf{x}_0)$ for $i = 1, ..., N-1$.
   (b) Set $\mathbf{x}_0^N = \widetilde{\mathbf{x}}_0$.
   (c) For $i = 1, ..., N$, set equal weights $\mathbf{w}_0^i$ normalised to sum to 1.
2. **For** $t = 1, ..., T$ **do:**
   (a) Draw $\mathbf{a}_t^i$ with $\mathrm{P}(\mathbf{a}_t^i = j) = \mathbf{w}_{t-1}^j$ for $i = 1, ..., N-1$.
   (b) Draw $\mathbf{x}_t^i \sim p(\mathbf{x}_t \mid \boldsymbol{\theta}, \mathbf{x}_{0:t-1}^{\mathbf{a}_t^i})$ for $i = 1, ..., N-1$.
   (c) Draw $\mathbf{a}_t^N$ with $\mathrm{P}(\mathbf{a}_t^N = j) \propto \mathbf{w}_{t-1}^j \, p(\widetilde{\mathbf{x}}_{t:T} \mid \boldsymbol{\theta}, \mathbf{x}_{1:t-1}^j)$.
   (d) Set $\mathbf{x}_t^N = \widetilde{\mathbf{x}}_t$.
   (e) For $i = 1, ..., N$, set $\mathbf{w}_t^i \propto p(\mathbf{y}_t \mid \boldsymbol{\theta}, \mathbf{x}_t^i)$, where the weights are normalised to sum to 1.

nel $M_{\boldsymbol{\theta}}^N$. However, it has been experienced in practice that the autocorrelation drops sharply as N increases (Lindsten et al., 2014), and for many models a moderate N is enough to obtain a rapidly mixing kernel.

A.2 VARIATIONAL BAYES

Variational inference (Jordan et al., 1999) is a popular method for approximate Bayesian inference based on making assumptions about the posterior over latent variables. In variational inference, the problem of solving intractable integrals is translated into the optimisation of a cost function.

We begin by expressing the log marginal likelihood in terms of the joint distribution over all variables and the posterior

$$\log p(y) = \log \frac{p(x, y)}{p(x|y)}. \tag{A.2}$$

Taking expectations on both sides with respect to an arbitrary distribution $q(x)$

$$\begin{aligned}
\log p(y) &= \langle \log \frac{p(x,y)}{p(x|y)} \rangle_{q(x)} && \text{(A.3a)} \\
&= \langle \log \frac{p(x,y)\, q(x)}{p(x|y)\, q(x)} \rangle_{q(x)} && \text{(A.3b)} \\
&= \langle \log \frac{p(x,y)}{q(x)} + \log \frac{q(x)}{p(x|y)} \rangle_{q(x)} && \text{(A.3c)} \\
&= \underbrace{\langle \log \frac{p(x,y)}{q(x)} \rangle_{q(x)}}_{\mathcal{L}(q(x))} + \underbrace{\langle \log \frac{q(x)}{p(x|y)} \rangle_{q(x)}}_{\mathrm{KL}(q(x)||p(x|y))}. && \text{(A.3d)}
\end{aligned}$$

Since a KL divergence is always greater or equal to zero (Murphy, 2012), we have

$$\log p(y) \geq \mathcal{L}(q(x)). \tag{A.4}$$

The bound is tight if and only if $q(x) = p(x|y)$. In other words, if the variational distribution $q(x)$ perfectly matches the posterior. Minimisation of the KL divergence between the variational distribution and the true posterior can be achieved by maximising the lower bound $\mathcal{L}(q(x))$. There are two main approaches to maximise the lower bound when the posterior is not analytically tractable:

1. Choose a parametric form for the variational distribution, parameterise it with θ and maximise the evidence lower bound with respect to those parameters
$$\theta^* = \arg\max_{\theta} \mathcal{L}(q_\theta(x)). \tag{A.5}$$

2. Choose a particular form for the approximate posterior (e.g. $q(x_1, x_2) = q(x_1)q(x_2)$) and use the calculus of variations to find the optimal distributions that maximise the lower bound. In the calculus of variations optimisation is done with respect to functions rather than parameters. In our case, the functions are probability distributions and need to satisfy the constraint that they integrate to one. Finding a stationary point of the evidence lower bound with respect to a variational distribution can be translated to the problem of finding the distribution that is a solution to the Euler-Lagrange differential equation. For
$$\mathcal{L}(q(x)) = \langle \log \frac{p(x,y)}{q(x)} \rangle_{q(x)} = \int I\left(x, q(x)\right) \, \mathrm{d}x, \tag{A.6}$$
the Euler-Lagrange equation is
$$\frac{\partial}{\partial q(x)} I\left(x, q(x)\right) = 0, \tag{A.7}$$
$$\frac{\partial}{\partial q(x)} q(x) \log \frac{p(x,y)}{q(x)} = 0, \tag{A.8}$$
$$\log \frac{p(x,y)}{q^*(x)} - 1 = 0. \tag{A.9}$$
Which results in the optimal variational distribution $q^*(x) \propto p(x,y)$ and since $p(x,y) \propto p(x|y)$ we obtain the result that was expected: the evidence lower bound is maximised when the variational distribution is equal to the true posterior. In such a case the bound is tight: $\log p(y) = \mathcal{L}(q^*(x))$. However, this is not particularly useful since the posterior is the very distribution that we are trying to find. It is by making assumptions about the variational distribution, such as $q(x_1, x_2) = q(x_1)q(x_2)$, that one can obtain useful results. This can be achieved by using the Euler-Lagrange equation to find stationary points of $\mathcal{L}$ with respect to parts of the variational distribution such as $q(x_1)$. The evidence lower bound is then maximised by executing a "coordinate"-wise optimisation in the space of variational distributions. Bishop (2006) and Murphy (2012) provide a much more thorough exposition of the calculus of variations and variational inference.

Bibliography

Agarwal, D. K. and Gelfand, A. E. (2005). Slice sampling for simulation based fitting of spatial data models. *Statistics and Computing*, 15(1):61–69. (cited in p. 40)

Alvarez, M., Luengo, D., and Lawrence, N. (2009). Latent force models. *Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics*, 5:9–16. (cited in p. 23)

Alvarez, M., Luengo, D., and Lawrence, N. (2013). Linear latent force models using Gaussian processes. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 35(11):2693–2705. (cited in p. 23)

Andrieu, C., Doucet, A., and Holenstein, R. (2010). Particle Markov chain Monte Carlo methods. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72(3):269–342. (cited in pp. 38, 39, 46, 62, and 87)

Andrieu, C., Moulines, E., and Priouret, P. (2005). Stability of stochastic approximation under verifiable conditions. *SIAM Journal on Control and Optimization*, 44(1):283–312. (cited in p. 46)

Andrieu, C. and Vihola, M. (2011). Markovian stochastic approximation with expanding projections. arXiv.org, arXiv:1111.5421. (cited in p. 46)

Barber, D., Cemgil, A. T., and Chiappa, S. (2011). *Bayesian Time Series Models*. Cambridge University Press. (cited in pp. 1 and 37)

Barber, D. and Chiappa, S. (2006). Unified inference for variational Bayesian linear Gaussian state-space models. In *Advances in Neural Information Processing Systems (NIPS)*, volume 20. (cited in p. 13)

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. Springer. (cited in pp. 45, 87, and 90)

Blocker, A. W. and Meng, X.-L. (2013). The potential and perils of preprocessing: Building new foundations. *Bernoulli*, 19(4):1176–1211. (cited in p. 76)

Box, G. E. P., Jenkins, G. M., and Reinsel, G. C. (1994). *Time Series Analysis: Forecasting and Control*. Prentice-Hall. (cited in p. 13)

Broderick, T., Boyd, N., Wibisono, A., Wilson, A. C., and Jordan, M. I. (2013). Streaming variational Bayes. In Burges, C., Bottou, L., Welling, M., Ghahramani, Z., and Weinberger, K., editors, *Advances in Neural Information Processing Systems 26*, pages 1727–1735. Curran Associates, Inc. (cited in p. 66)

Chen, T., Ohlsson, H., and Ljung, L. (2012). On the estimation of transfer functions, regularizations and Gaussian processesrevisited. *Automatica*, 48(8):1525–1535. (cited in p. 4)

Damianou, A. and Lawrence, N. (2013). Deep Gaussian processes. In Carvalho, C. and Ravikumar, P., editors, *Proceedings of the Sixteenth International Workshop on Artificial Intelligence and Statistics (AISTATS-13)*, AISTATS '13, pages 207–215. JMLR W&CP 31. (cited in p. 22)

Damianou, A. and Lawrence, N. (2015). Semi-described and semi-supervised learning with Gaussian processes. *Uncertainty in Artificial Intelligence (UAI)*. (cited in pp. 21 and 77)

Damianou, A. C., Titsias, M., and Lawrence, N. D. (2011). Variational Gaussian process dynamical systems. In Shawe-Taylor, J., Zemel, R., Bartlett, P., Pereira, F., and Weinberger, K., editors, *Advances in Neural Information Processing Systems 24*, pages 2510–2518. (cited in pp. 22 and 34)

Deisenroth, M. (2010). *Efficient Reinforcement Learning using Gaussian Processes*. PhD thesis, Karlsruher Institut für Technologie. (cited in p. 42)

Deisenroth, M. and Mohamed, S. (2012). Expectation Propagation in Gaussian process dynamical systems. In Bartlett, P., Pereira, F., Burges, C., Bottou, L., and Weinberger, K., editors, *Advances in Neural Information Processing Systems 25*, pages 2618–2626. (cited in p. 20)

Deisenroth, M., Turner, R., Huber, M., Hanebeck, U., and Rasmussen, C. (2012). Robust filtering and smoothing with Gaussian processes. *Automatic Control, IEEE Transactions on*, 57(7):1865 –1871. (cited in p. 20)

Delyon, B., Lavielle, M., and Moulines, E. (1999). Convergence of a stochastic approximation version of the EM algorithm. *The Annals of Statistics*, 27(1):pp. 94–128. (cited in p. 46)

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):pp. 1–38. (cited in p. 45)

Doucet, A. and Johansen, A. (2011). A tutorial on particle filtering and smoothing: Fifteen years later. In Crisan, D. and Rozovsky, B., editors, *Oxford Hand-book of Nonlinear Filtering*. Oxford University Press. (cited in p. 39)

Duchi, J., Hazan, E., and Singer, Y. (2011). Stochastic variational inference. *Journal of Machine Learning Research*, 12:2121–2159. (cited in p. 63)

Duvenaud, D. (2014). *Automatic Model Construction with Gaussian Processes*. PhD thesis, Computational and Biological Learning Laboratory, University of Cambridge. (cited in p. 78)

Duvenaud, D., Lloyd, J., Grosse, R., Tenenbaum, J., and Zoubin, G. (2013). Structure discovery in nonparametric regression through compositional kernel search. In *Proceedings of The 30th International Conference on Machine Learning (ICML)*, pages 1166–1174. (cited in p. 13)

Ferris, B., Fox, D., and Lawerence, N. (2007). WiFi-SLAM using Gaussian process latent variable models. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*. (cited in p. 18)

Foti, N., Xu, J., Laird, D., and Fox, E. (2014). Stochastic variational inference for hidden markov models. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N., and Weinberger, K., editors, *Advances in Neural Information Processing Systems 27*, pages 3599–3607. Curran Associates, Inc. (cited in p. 66)

Frigola, R., Chen, Y., and Rasmussen, C. E. (2014a). Variational Gaussian process state-space models. In *Advances in Neural Information Processing Systems 27 (NIPS)*, pages 3680–3688. (cited in pp. 55 and 66)

Frigola, R., Lindsten, F., Schön, T. B., and Rasmussen, C. E. (2013). Bayesian inference and learning in Gaussian process state-space models with particle MCMC. In Bottou, L., Burges, C., Ghahramani, Z., Welling, M., and Weinberger, K., editors, *Advances in Neural Information Processing Systems 26 (NIPS)*, pages 3156–3164. (cited in pp. 16, 37, 53, and 72)

Frigola, R., Lindsten, F., Schön, T. B., and Rasmussen, C. E. (2014b). Identification of Gaussian process state-space models with particle stochastic approximation EM. In *19th World Congress of the International Federation of Automatic Control (IFAC)*, pages 4097–4102. (cited in pp. 17 and 37)

Frigola, R. and Rasmussen, C. E. (2013). Integrated pre-processing for Bayesian nonlinear system identification with Gaussian processes. In *Decision and Control (CDC), 2013 IEEE 52nd Annual Conference on*, pages 5371–5376. (cited in pp. 21 and 75)

Gal, Y. and Ghahramani, Z. (2015). Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *Deep Learning Workshop, ICML*. (cited in p. 86)

Ghahramani, Z. (2012). Bayesian nonparametrics and the probabilistic approach to modelling. *Philosophical Transactions of the Royal Society A*. (cited in pp. 4 and 45)

Ghahramani, Z. and Roweis, S. (1999). Learning nonlinear dynamical systems using an EM algorithm. In M. J. Kearns, S. A. S. and Cohn, D. A., editors, *Advances in Neural Information Processing Systems 11*. MIT Press. (cited in pp. 20, 26, and 68)

Gill, P., Golub, G., Murray, W., and Saunders, M. (1974). Methods for modifying matrix factorizations. *Mathematics of Computation*, 126(28):505–535. (cited in p. 53)

Girard, A., Rasmussen, C. E., Quiñonero-Candela, J., and Murray-Smith, R. (2003). Gaussian process priors with uncertain inputs — application to multiple-step ahead time series forecasting. In Becker, S., Thrun, S., and Obermayer, K., editors, *Advances in Neural Information Processing Systems 15*, pages 529–536, Cambridge, MA, USA. The MIT Press. (cited in p. 14)

Gordon, N., Salmond, D., and Smith, A. (1993). Novel approach to nonlinear/non-gaussian Bayesian state estimation. *Radar and Signal Processing, IEE Proceedings F*, 140(2):107–113. (cited in p. 41)

Graves, A., Mohamed, A.-R., and Hinton, G. (2013). Speech recognition with deep recurrent neural networks. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pages 6645–6649. (cited in p. 86)

Gregorcic, G. and Lightbody, G. (2002). Gaussian processes for modelling of dynamic non-linear systems. In *Proceedings of the Irish Signals and Systems Conference*, pages 141–147. (cited in p. 14)

Grimmett, G. and Stirzaker, D. (2001). *Probability and Random Processes*. Oxford University Press. (cited in p. 13)

Gutjahr, T., Ulmer, H., and Ament, C. (2012). Sparse Gaussian processes with uncertain inputs for multi-step ahead prediction. In *Proceedings of the 16th IFAC Symposium on System Identification (SYSID)*, volume 16, pages 107–112. (cited in p. 15)

Hamilton, J. D. (1994). *Time Series Analysis*. Princeton University Press. (cited in p. 1)

Hensman, J., Fusi, N., and Lawrence, N. D. (2013). Gaussian processes for big data. *Conference on Uncertainty in Artificial Intelligence, (UAI)*. (cited in pp. 33 and 34)

Hensman, J., Matthews, A., and Ghahramani, Z. (2015). Scalable variational Gaussian process classification. *18th International Conference on Artificial Intelligence and Statistics, (AISTATS)*. (cited in pp. 33 and 34)

Hinton, G. (2012). Lecture 6: Stochastic optimisation. *Coursera: Neural Networks for Machine Learning*. (cited in p. 63)

Hjalmarsson, H., Rojas, C. R., and Rivera, D. E. (2012). System identification: A Wiener-Hammerstein benchmark. *Control Engineering Practice*, 20(11):1095 – 1096. (cited in p. 81)

Hoffman, M. D., Blei, D. M., Wang, C., and Paisley, J. (2013). Stochastic variational inference. *Journal of Machine Learning Research*, 14:1303–1347. (cited in p. 65)

Isermann, R. and Münchhof, M. (2011). *Identification of Dynamic Systems, An Introduction with Applications*. Springer. (cited in pp. 3 and 4)

Izhikevich, E. M. (2000). Neural excitability, spiking and bursting. *International Journal of Bifurcation and Chaos*, 10(06):1171–1266. (cited in p. 73)

Jaynes, E. T. (2003). *Probability Theory, The Logic of Science*. Cambridge University Press. (cited in p. 2)

Jordan, M. I. (2013). On statistics, computation and scalability. *Bernoulli*, 19:1378–1390. (cited in p. 76)

Jordan, M. I., Ghahramani, Z., Jaakkola, T. S., and Saul, L. K. (1999). An introduction to variational methods for graphical models. *Machine Learning*, 37(2):183–233. (cited in pp. 56 and 89)

Kingma, D. P. and Ba, J. L. (2015). Adam: A method for stochastic optimization. *International Conference on Learning Representations (ICLR 2015)*. (cited in p. 63)

Ko, J. and Fox, D. (2011). Learning GP-BayesFilters via Gaussian process latent variable models. *Autonomous Robots*. (cited in p. 20)

Kocijan, J., Girard, A., Banko, B., and Murray-Smith, R. (2003). Dynamic systems identification with Gaussian processes. *IEEE Region 8 EUROCON: computer as a tool*, A:352–356. (cited in p. 14)

Lawrence, N. D. (2005). Probabilistic non-linear principal component analysis with gaussian process latent variable models. *Journal of Machine Learning Research*, 6:1783–1816. (cited in p. 22)

Lawrence, N. D. and Moore, A. J. (2007). Hierarchical Gaussian process latent variable models. In *Proceedings of the 24th International Conference on Machine learning*, pages 481–488. (cited in p. 22)

Lázaro-Gredilla, M., Quiñonero-Candela, J., Rasmussen, C. E., and Figueiras-Vidal, A. R. (2010). Sparse spectrum Gaussian process regression. *Journal of Machine Learning Research*, 11:1865–1881. (cited in p. 69)

LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *Nature*, 521:436–444. (cited in pp. 34, 67, and 86)

Lindsten, F. (2013). An efficient stochastic approximation EM algorithm using conditional particle filters. In *Proceedings of the 38th IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 6274–6278. (cited in p. 46)

Lindsten, F., Jordan, M. I., and Schön, T. B. (2014). Particle Gibbs with ancestor sampling. *Journal of Machine Learning Research*, 15:2145–2184. (cited in pp. 39, 46, 53, 87, 88, and 89)

Lindsten, F. and Schön, T. B. (2013). Backward simulation methods for Monte Carlo statistical inference. *Foundations and Trends in Machine Learning*, 6(1):1–143. (cited in pp. 39 and 47)

Ljung, L. (1999). *System Identification: Theory for the User*. Prentice Hall, second edition. (cited in pp. 1, 2, 7, 13, and 76)

Ljung, L. (2012). *System Identification Toolbox^{TM}User's Guide (R2012b)*. The MathWorks. (cited in pp. 81 and 83)

MacKay, D. J. C. (2003). *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press. (cited in pp. 3, 11, 45, and 59)

Matthews, A. G. d. G., Hensman, J., Turner, R. E., and Ghahramani, Z. (2015). On sparse variational methods and the Kullback-Leibler divergence between stochastic processes. *arXiv:1504.07027*. (cited in p. 31)

McHutchon, A. (2014). *Nonlinear Modelling and Control using Gaussian Processes*. PhD thesis, University of Cambridge. (cited in pp. 17, 20, 38, 39, 47, and 68)

McHutchon, A. and Rasmussen, C. E. (2011). Gaussian process training with input noise. In *Advances in Neural Information Processing Systems 25*, Granada, Spain. (cited in pp. 21 and 77)

McHutchon, A. and Rasmussen, C. E. (2014). Comparing learning in Gaussian process state space models. *submitted*. (cited in p. 17)

Micchelli, C. A., Xu, Y., and Zhang, H. (2006). Universal kernels. *Journal of Machine Learning Research*, 7:2651–2667. (cited in p. 26)

Mizuseki, K., Sirota, A., Pastalkova, E., Diba, K., and Buzski, G. (2013). Multiple single unit recordings from different rat hippocampal and entorhinal regions while the animals were performing multiple behavioral tasks. CRCNS.org. `http://dx.doi.org/10.6080/K09G5JRZ`. (cited in p. 72)

Murphy, K. P. (2012). *Machine Learning, a Probabilistic Perspective*. MIT Press. (cited in pp. 37, 89, and 90)

Murray, I., Adams, R. P., and MacKay, D. J. (2010). Elliptical slice sampling. *JMLR: W&CP*, 9:541–548. (cited in p. 11)

Neal, R. M. (2003). Slice sampling. *Ann. Statist.*, 31(3):705–767. (cited in p. 40)

Neal, R. M. and Hinton, G. E. (1998). A view of the EM algorithm that justifies incremental, sparse, and other variants. In Jordan, M. I., editor, *Learning in Graphical Models*, volume 89 of *NATO ASI Series*, pages 355–368. Springer. (cited in p. 65)

Nguyen, T. V. and Bonilla, E. V. (2014). Automated variational inference for Gaussian process models. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N., and Weinberger, K., editors, *Advances in Neural Information Processing Systems 27*, pages 1404–1412. (cited in p. 11)

Opper, M. (1998). A Bayesian approach to on-line learning. In Saad, D., editor, *On-Line Learning in Neural Networks*. Cambridge University Press. (cited in p. 66)

Orbanz, P. (2014). Lecture notes on Bayesian nonparametrics. *Columbia University*. (cited in p. 4)

Overschee, P. V. and Moor, B. D. (1996). *Subspace Identification for Linear Systems: Theory Implementation Applications*. Kluwer. (cited in pp. 13 and 72)

Peterka, V. (1981). Bayesian system identification. *Automatica*, 17(1):41 – 53. (cited in pp. 2 and 4)

Quiñonero-Candela, J., Girard, A., Larsen, J., and Rasmussen, C. E. (2003). Propagation of uncertainty in Bayesian kernel models - application to multiple-step ahead forecasting. In *ICASSP 2003*, volume 2, pages 701–704. (cited in p. 72)

Quiñonero-Candela, J. and Rasmussen, C. E. (2005). A unifying view of sparse approximate Gaussian process regression. *Journal of Machine Learning Research*, 6:1939–1959. (cited in pp. 33, 34, 51, 52, 81, and 83)

Rasmussen, C. E. and Williams, C. K. I. (2006). *Gaussian Processes for Machine Learning*. The MIT Press. (cited in pp. 5, 8, 9, 11, 19, 27, 48, 49, 51, and 83)

Robbins, H. and Monro, S. (1951). A stochastic approximation method. *Ann. Math. Statist.*, 22(3):400–407. (cited in p. 46)

Roberts, S., Osborne, M., Ebden, M., Reece, S., Gibson, N., and Aigrain, S. (2012). Gaussian processes for time-series modelling. *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 371(1984). (cited in p. 13)

Särkkä, S. (2013). *Bayesian Filtering and Smoothing*. Cambridge University Press. (cited in pp. 62 and 66)

Schoukens, J., Suykens, J., and Ljung, L. (2009). Wiener-Hammerstein benchmark. In *Symposium on System Identification (SYSID) Special Session*, volume 15. (cited in p. 81)

Shalizi, C. (2008). Book review: "Hidden Markov models and dynamical systems" by Andrew M. Fraser. *The Bactra Review, http://bactra.org/reviews/fraser-on-HMMs/*, (139). (cited in p. 25)

Shumway, R. H. and Stoffer, D. S. (1982). An approach to time series smoothing and forecasting using the EM algorithm. *Journal of Time Series Analysis*, 3(4):253–264. (cited in pp. 13 and 26)

Shumway, R. H. and Stoffer, D. S. (2011). *Time Series Analysis and Its Applications*. Springer. (cited in pp. 1 and 2)

Snelson, E. and Ghahramani, Z. (2006). Sparse Gaussian processes using pseudo-inputs. In Weiss, Y., Schölkopf, B., and Platt, J., editors, *Advances in Neural Information Processing Systems 18*, pages 1257–1264. The MIT Press, Cambridge, MA. (cited in pp. 51, 52, and 80)

Solin, A. and Särkkä, S. (2014). Hilbert space methods for reduced-rank Gaussian process regression. *arXiv preprint arXiv:1401.5508*. (cited in p. 69)

Spiegelhalter, D. and Rice, K. (2009). Bayesian statistics. *Scholarpedia*, 4(3):5230. (cited in p. 3)

Sutskever, I. (2013). *Training Recurrent Neural Networks*. PhD thesis, University of Toronto. (cited in pp. 34, 67, and 86)

Svensson, A., Solin, A., Särkkä, S., and Schön, T. B. (2015). Computationally efficient Bayesian learning of Gaussian process state space models. *arXiv:1506.02267*. (cited in p. 69)

Titsias, M. and Lawrence, N. (2010). Bayesian Gaussian process latent variable model. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 9, pages 844–851. (cited in pp. 22 and 34)

Titsias, M. K. (2009). Variational learning of inducing variables in sparse Gaussian processes. *Twelfth International Conference on Artificial Intelligence and Statistics, (AISTATS)*, pages 567–574. (cited in pp. 33, 34, 52, 55, 56, 59, 63, 80, and 81)

Titsias, M. K. and Lázaro-Gredilla, M. (2014). Doubly stochastic variational Bayes for non-conjugate inference. *31st International Conference on Machine Learning (ICML)*. (cited in p. 66)

Tsay, R. S. (2013). *Multivariate Time Series Analysis*. Wiley. (cited in p. 1)

Turner, R. D. (2011). *Gaussian Processes for State Space Models and Change Point Detection*. PhD thesis, University of Cambridge. (cited in p. 13)

Turner, R. D., Deisenroth, M. P., and Rasmussen, C. E. (2010). State-space inference and learning with Gaussian processes. In Teh, Y. W. and Titterington, M., editors, *13th International Conference on Artificial Intelligence and Statistics*, volume 9 of *W&CP*, pages 868–875, Chia Laguna, Sardinia, Italy. Journal of Machine Learning Research. (cited in pp. 17, 20, and 68)

Turner, R. E., Rasmussen, C. E., van der Wilk, M., Bui, T. D., and Frigola, R. (2015). Gaussian process state space models. Unpublished Note, January 2015. (cited in p. 30)

Turner, R. E. and Sahani, M. (2011). Two problems with variational expectation maximisation for time-series models. In Barber, D., Cemgil, T., and Chiappa, S., editors, *Bayesian Time Series Models*, chapter 5, pages 109–130. Cambridge University Press. (cited in p. 59)

Wang, J., Fleet, D., and Hertzmann, A. (2006). Gaussian process dynamical models. In Weiss, Y., Schölkopf, B., and Platt, J., editors, *Advances in Neural Information Processing Systems 18*, pages 1441–1448. MIT Press, Cambridge, MA. (cited in pp. 19, 20, and 29)

Wang, J., Fleet, D., and Hertzmann, A. (2008). Gaussian process dynamical models for human motion. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 30(2):283–298. (cited in pp. 19 and 20)

Wu, Y., Hernandez-Lobato, J. M., and Ghahramani, Z. (2014). Gaussian process volatility model. *arXiv:1402.3085v1 [stat.ME]*. (cited in p. 17)