Probabilistic Methods of Parameter Inference for Differential Equations

by

Mohan Wu

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Statistics

Waterloo, Ontario, Canada, 2025

© Mohan Wu 2025

Examining Committee Membership

The following served on the Examining Committee for this thesis. The decision of the Examining Committee is by majority vote.

External Examiner:	Edward Ionides Professor, Dept. of Statistics, University of Michigan
Supervisor:	Martin Lysy Associate Professor, Dept. of Statistics and Actuarial Science, University of Waterloo
Internal Members:	Samuel Wong Associate Professor, Dept. of Statistics and Actuarial Science, University of Waterloo
	Jeffrey Negrea Assistant Professor, Dept. of Statistics and Actuarial Science, University of Waterloo
Internal-External Member:	Roberto Guglielmi Assistant Professor, Dept. of Applied Mathematics, University of Waterloo

Author's Declaration

This thesis consists of material all of which I authored or co-authored: see Statement of Contributions included in the thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Statement of Contributions

I am the sole author of Chapter 1, 4, and 5.

Chapter 2 is based on a manuscript that has been submitted to the *Journal of Statistical Software*. I am the first author of that manuscript, and the associated software:

- Wu, M., Lysy, M. (2025). rodeo: Probabilistic methods of parameter inference for ordinary differential equations. https://arxiv.org/abs/2506.21776.
- Wu, M., Lysy, M. (2025). rodeo: Probabilistic methods of parameter inference for ODEs. https://pypi.org/project/rodeo.

Chapter 3 is based on the manuscript published in *27th International Conference on Artificial Intelligence and Statistics*. I am the first author of that manuscript:

- Wu, M., Lysy, M. (2024). Data-adaptive probabilistic likelihood approximation for ordinary differential equations. Proceedings of The 27th International Conference on Artificial Intelligence and Statistics, volume 238 of Proceedings of Machine Learning Research, pages 1018–1026.PMLR, 02–04 May 2024. https://proceedings.mlr.press/v238/wu24b.html.

The associated software is available in the **rodeo** github.

Chapter 4 is based on original research that is intended to form the basis of a future submission. The associated software is available at:

- Wu, M., Lysy, M. (2025) IEVI: Information-Efficient Variational Inference for SDEs. https://github.com/mlysy/sde-ievi.

Abstract

Parameter estimation for differential equations is a fundamental problem in many scientific fields. This thesis is concerned with developing statistically and computationally efficient methods of parameter inference for ordinary differential equations (ODEs) and stochastic differential equations (SDEs). For ODEs, this often involves numerically solving the differential equation at each likelihood evaluation, a task traditionally handled with deterministic numerical solvers. Here we consider likelihood approximations based on probabilistic numerical solvers, which have been shown to produce more reliable parameter estimates by better accounting for numerical uncertainty. In contrast, parameter inference for SDEs requires integrating over high-dimensional latent variables in state-space models, which is computationally expensive. We propose a variational inference framework that significantly reduces this computational burden by converting the integration problem into an optimization problem.

Chapter 2 is a review of several existing probabilistic ODE solvers and associated parameter inference methods. In particular, we detail the commonly-used paradigm of approximating the ODE solution with a nonlinear state-space model, and then linearizing it to easily perform relevant computations via the Kalman filtering and smoothing recursions. In the data-free setting, we extend convergence results previously established only for the forward pass of the Kalman algorithm to the backward pass as well. This is a key result for establishing the convergence of several associated parameter learning methods. We provide empirical evidence that supports our theory and demonstrate that the backward pass estimator is more accurate than using a forward pass alone. We also propose a novel variation of a probabilistic method for parameter inference.

In Chapter 3, we present a novel probabilistic approximation to the ODE likelihood that reduces the parameter sensitivity inherent in the true likelihood by directly learning from noisy observations. Leveraging the efficient Kalman filter algorithm, our method scales linearly in both ODE variables and time discretization points. Furthermore, it is applicable to ODE problems with partially unobserved components and arbitrary measurement noise. Several numerical experiments demonstrate that our method produces more reliable estimates when compared to other probabilistic methods and, in extremely sensitive problems, exhibits greater robustness than the exact ODE likelihood.

In Chapter 4, we introduce a scalable stochastic variational inference framework for estimating parameters in SDE models. Our method effectively captures the forward-backward information propagation in state-space models by using a recurrent neural network (RNN) to estimate the quantities needed for the forward-backward recursions. The procedure

scales linearly with the number of SDE discretization steps. Experimental results show that it produces more reliable parameter posteriors than a number of competing variational and non-variational methods, particularly in high-dimensional random effects models.

Acknowledgements

I would first like to thank my supervisor, Dr. Martin Lysy, for his constant support throughout my graduate studies. It has been seven years since I first began working with him as an undergraduate, and throughout that time, he has been incredibly patient and generous with his time. Whether it was reviewing a manuscript or debugging code, he was always quick to set up a meeting to provide thoughtful feedback. I am especially grateful to Martin for teaching me how to approach difficult research problems systematically. His guidance has taught me that meaningful progress often comes from persistent, incremental work and the ingenuity to keep chipping away at the problem, little by little.

I would like to thank my committee members, Dr. Edward Ionides, Dr. Samuel Wong, Dr. Jeffery Negrea, and Dr. Roberto Guglielmi, for generously dedicating their time to attend my defence and for providing valuable feedback that improved the clarity of this thesis. A special thanks to Dr. Samuel Wong and Dr. Jeffery Negrea for also attending my thesis proposal.

I would also like to thank my friends, Luke Hagar, Alexandra Mossman, and Minzee Kim, for accompanying me on the PhD journey. Since Luke graduated before the rest of us, I am especially grateful to him for always being willing to help whenever I had questions about the PhD process. I am also thankful to my research lab members, Pranav Subramani, Jonathan Ramkissoon, Michelle Ko, and Ferris Zhu, from whom I have learned a great deal about programming, mathematics, and many other topics.

Lastly, I would like to thank my parents for their support and for providing me with opportunities they themselves never had growing up. Their sacrifices allowed me to pursue my education in comfort and gave me the confidence to push myself further. I am especially grateful to my sister, who was the first to encourage me to consider pursuing a PhD.

Dedication

This thesis is dedicated to my parents.

Table of Contents

Examining Committee Membership **ii**

Author's Declaration **iii**

Statement of Contributions **iv**

Abstract **v**

Acknowledgements **vii**

Dedication **viii**

List of Figures **xiii**

List of Tables **xv**

List of Abbreviations **xvi**

1 Introduction **1**

2 Bayesian Filtering Probabilistic Solvers for Ordinary Differential Equations **5**

2.1 Introduction 5

2.2 State-Space Model Representing the Initial Value Problem 7

2.2.1 Gaussian Markov Process Prior 7
2.2.2 Model Interrogations 8
2.3 Kalman Filtering and Smoothing 10
2.3.1 Convergence of the Kalman Filter Estimates 13
2.3.2 Convergence of the RTS Smoother Estimates 15
2.4 Numerical Experiments on Convergence Rates 16
2.4.1 FitzHugh-Nagumo Model 16
2.5 Parameter Inference for Gaussian Filtering ODE Solvers 19
2.5.1 Basic Method 21
2.5.2 Physics-Enhanced Regression in Initial Value problems 21
2.5.3 Marginal MCMC 23
2.5.4 MAGI 24
2.6 Numerical Experiments on Parameter Inference 28
2.6.1 Parameter Posteriors for the Three Methods 30
2.7 Discussion 32

3 Data-Adaptive Probabilistic Likelihood Approximation For Ordinary Differential Equations 35
3.1 Introduction 35
3.2 Data-Adaptive Probabilistic ODE Likelihood Approximation 36
3.2.1 Gaussian Observation Model 38
3.2.2 Non-Gaussian Observation Model 39
3.3 Numerical Experiments on Parameter Inference 42
3.3.1 SEIRAH model 43
3.3.2 Lorenz63 model 45
3.4 Discussion 50

4 An Information-Efficient Variational Inference Method for Stochastic Differential Equations **52**

4.1 Introduction 52

4.2 Background 55

4.2.1 Stochastic Differential Equations 55

4.2.2 Particle Filtering 56

4.2.3 Laplace Approximation 57

4.2.4 Max-and-Smooth 58

4.2.5 Variational Inference 59

4.3 Information-Efficient Variational Inference 61

4.3.1 Latent Variational Distribution 61

4.3.2 Linear Gaussian Observation Model 64

4.3.3 Higher Resolution 64

4.4 Related Works 65

4.5 Numerical Experiments 67

4.5.1 Lotka–Volterra Model 68

4.5.2 Stochastic Volatility 72

4.5.3 Orange Tree Growth Model 77

4.6 Discussion 83

5 Conclusion **85**

References **87**

APPENDICES **100**

A Additional Material for Chapter 2 **101**

A.1 Proofs for Lemmas and Theorems 101

A.2 Kalman recursions 106

B Additional Material for Chapter 3 **108**

B.1 DALTON for Non-Gaussian Observations 108

B.2 Parameter estimation experiment details 111

B.3 Lorenz63 Model Extra Figures 111

List of Figures

2.1 The true solution of the FitzHugh-Nagumo (FN) model. 17

2.2 Convergence rates for the Kalman updated mean and the Rauch-Tung-Striebel (RTS) smoothed mean using independent $\{1, 2\}$-times integrated Brownian motion (IBM) prior. 18

2.3 Convergence rates for the Kalman updated variance and the RTS smoothed variance using independent $\{1, 2\}$-times IBM prior. 20

2.4 Simulated ODE and noisy observations of the FN model. 29

2.5 Parameter posteriors for the FN model using the Bayesian normal approximation for the Euler ODE solver, the Basic ODE likelihood approximation, and Physics-enhanced regression in initial value problems (Fenrir). Also included are posteriors from the marginal Markov chain Monte Carlo (MCMC) method, as well as Markov MAnifold-constrained Gaussian process Inference (MAGI) sampled via MCMC. 33

3.1 Parameter posteriors for the SEIRAH model using various ODE solvers. . 46

3.2 $x(t)$ for the Lorenz63 model calculated using various solvers. 47

3.3 Parameter posteriors for the Lorenz63 model using various ODE solvers at two initial parameter values. 48

3.4 Initial and optimized parameter values for the Lorenz63 model using various ODE solvers. 50

3.5 ODE posteriors of $x(t)$ for the Lorenz63 model using data-adaptive probabilistic ODE likelihood approximation (DALTON) with $\Delta t = 0.00125$. 51

4.1 Unrolled RNN from time step 0 to m, where each cell updates the hidden state using the current observation and produces corresponding outputs. . 62

4.2 Illustration of resolution $r = 3$, where $\Delta t = \Delta t'/3$. 65

4.3 $S = 1000$ sample draws from the variational distributions of information-efficient variational inference (IEVI) and the method of Ryder et al. (2018), and $P = 100,000$ particles from particle filtering, shown alongside the ground truth latent trajectory and observed data. 70

4.4 Parameter posteriors for the Lotka-Volterra model using various methods. 71

4.5 evidence lower bound (ELBO) convergence curves for training IEVI using various batch sizes of the observation sequence. 74

4.6 $S = 1000$ samples of the latent trajectories from the variational distributions produced by IEVI and the method of Ryder et al. (2018), as well as from MCMC post burn-in, shown alongside the ground truth latent trajectory and observed data. 75

4.7 Parameter posteriors for the stochastic volatility model using various methods. Some of the Ryder et al. (2018) posteriors have been clipped for visual clarity. The corresponding mean (standard deviation) estimates for the missing Ryder et al. (2018) posteriors are: $\hat{\alpha} = 1.97(0.03), \hat{\gamma} = 0.17(0.04)$, and $\hat{\mu} = 3.43(0.05)$. 76

4.8 $S = 1000$ sample draws from the variational distributions of IEVI and the method of Ryder et al. (2018), and $P = 10,000$ particles from particle filtering, shown alongside the ground truth latent trajectory and observed data for the orange tree growth model. Also included are 1000 MCMC samples post burn-in. 79

4.9 Parameter posteriors for the orange tree growth model using various methods. Some of the Laplace approximation posteriors have been clipped for visual clarity. The corresponding mean (standard deviation) estimates for the missing Laplace posteriors are: $\hat{\sigma}_{\eta_2} = 144(4.31)$, and $\hat{\sigma} = 0.18(0.0012)$. 83

B.1 The full latent variables for the Lorenz63 model calculated using various solvers. 112

B.2 ODE posteriors for the Lorenz63 model using DALTON. Extended version of Figure 3.5. 113

List of Tables

2.1 Various model interrogation methods. 10

4.1 The root mean squared error (RMSE) for each parameter of the Lotka-Volterra model across 100 simulations, using various inference methods. The number in brackets indicates how many times the true parameter value was contained within the 95% credible interval. 72

4.2 The RMSE for each parameter of the stochastic volatility model across 100 simulations, using various inference methods. The number in brackets indicates how many times the true parameter value was contained within the 95% credible interval. 77

4.3 The RMSE for each parameter of the orange tree growth model across 100 simulations, using various inference methods. The number in brackets indicates how many times the true parameter value was contained within the 95% credible interval. 82

List of Abbreviations

AD automatic differentiation 40, 56, 57

BFGS Broyden–Fletcher–Goldfarb–Shanno 66

DALTON data-adaptive probabilistic ODE likelihood approximation 3, 34–37, 39, 40, 42, 44, 45, 47–49, 82, 83, 93, 94

ELBO evidence lower bound 58, 59, 61, 63, 65, 71, 72

Fenrir Physics-enhanced regression in initial value problems 20–22, 28, 30, 31, 35, 36, 39, 40, 42, 44–47, 49

FN FitzHugh-Nagumo 15, 16, 26, 28, 30, 31

HMC Hybrid Monte Carlo 30, 76, 78

IBM integrated Brownian motion 6, 15–18, 32, 86, 87, 89, 95

IEVI information-efficient variational inference 2, 3, 52, 53, 59, 64, 65, 67–77, 79–81, 83

IVP initial value problem 4–6, 9, 13, 15, 16, 34, 41

JIT just-in-time 3, 5, 29, 40

KL Kullback-Leibler 58

MAE maximum absolute error 14

MAGI MAnifold-constrained Gaussian process Inference 3, 5, 27, 30, 31, 82

MCMC Markov chain Monte Carlo 1, 2, 7, 22, 25, 26, 28–31, 42, 51, 52, 57, 66, 72–80, 82, 83

MEV maximum error in variance 14

ODE ordinary differential equation v, 1–9, 11–16, 19, 21, 26–28, 30–36, 39–44, 46–50, 82, 83, 85, 95

RKDP Dormand-Prince 8/7 solver consisting of an 8th order Runge-Kutta method with 7th order step size adaptation 27, 30, 41, 42, 44–47

RMSE root mean squared error 69, 70, 74, 75, 79

RNN recurrent neural network v, 52, 59–63, 70, 71, 76–78, 80, 83

rodeo probabilistic ode solver 3

RTS Rauch-Tung-Striebel 5–7, 11, 14, 16–21, 30, 32, 39, 82, 85, 87–89

SDE stochastic differential equation v, vi, 2, 3, 50–53, 59, 61, 62, 66, 71, 80, 82, 83

SSM state-space model 2, 3, 5–8, 23, 47, 51–54, 56, 57, 59, 63, 83

Chapter 1

Introduction

Parameter estimation in time-series analysis is an important problem in the natural sciences, engineering, and economics. A common modelling approach is to assume that observations are noisy measurements of underlying latent variables which evolve dynamically over time. For applications in which these dynamics occur in continuous time, two prominent modelling classes are ordinary differential equations (ODEs) and stochastic differential equations (SDEs).

ODEs models are routinely used in applications for which the underlying latent process is taken to be a deterministic function of the initial conditions. To perform parameter inference, the ODE typically must be solved at each parameter value. However, these complex systems often lack closed-form solutions and must be approximated numerically. Until recently, ODE solutions were typically approximated using deterministic algorithms (e.g., Butcher, 2008; Griffiths and Higham, 2010; Atkinson et al., 2009), but a new research field called *probabilistic numerics* (Diaconis, 1988; Skilling, 1992; Hennig et al., 2015) has experimented with solving ODEs using probabilistic solvers. These probabilistic solvers have been noted to be more reliable in parameter inference problems than deterministic alternatives because the former directly and naturally account for uncertainty in the numerical approximation (Chkrebtii et al., 2016; Conrad et al., 2017).

Over the past decade, many probabilistic ODE algorithms have been proposed (e.g., Dondelinger et al., 2013; Barber and Wang, 2014; Schober et al., 2014; Chkrebtii et al., 2016; Kersting and Hennig, 2016; Teymur et al., 2016; Conrad et al., 2017; Ghosh et al., 2017; Gorbach et al., 2017; Lazarus et al., 2018; Tronarp et al., 2018; Schober et al., 2019; Wenk et al., 2019; Krämer and Hennig, 2020; Yang et al., 2021; Wenger et al., 2021; Bosch et al., 2021; Krämer et al., 2022; Tronarp et al., 2022; Bosch et al., 2022). Although theoretically

sound, most implementations are several times slower than deterministic ODE solvers and are primarily used on toy examples to serve as a proof-of-concept. This performance gap is expected, given that deterministic algorithms have benefited from over a century of optimization, whereas probabilistic methods are comparatively recent. Furthermore, most probabilistic algorithms require either large matrix operations or computationally expensive methods such as Markov chain Monte Carlo (MCMC) to integrate out the latent variables at each solution time step.

A noteworthy family of probabilistic solvers (Bosch et al., 2021; Krämer et al., 2022; Tronarp et al., 2022; Bosch et al., 2022) that utilize a Bayesian filtering paradigm based on the original paper by Tronarp et al. (2018) does not exhibit either of these weaknesses. These solvers employ a state-space model (SSM) for the ODE and apply the Kalman filter (Kalman, 1960) to approximate the solution. Notably, these solvers have demonstrated both higher accuracy and faster performance compared to the most popular deterministic solvers across numerous examples (Wenger et al., 2021).

Our research makes several contributions to the study of probabilistic ODE solvers. First, we extend a theoretical result of Kersting et al. (2020b) on convergence of an approximate solution to the Bayesian filtering paradigm to the true ODE solution. We also review and compare several parameter inference methods with probabilistic solvers from the literature (Chkrebtii et al., 2016; Yang et al., 2021; Tronarp et al., 2022), and introduce a novel computationally-efficient variant of one such method. In addition, we propose a new algorithm inspired by the approach of Tronarp et al. (2022), which effectively conditions on observations to reduce sensitivity to parameter initialization.

While ODEs models are ideal for systems assumed to evolve deterministically, many real-world phenomena exhibit inherent stochasticity, making SDEs a more appropriate modelling choice. Parameter inference for SDEs is considerably more challenging, as it requires integrating over high-dimensional latent variables. Since closed-form solutions are rarely available, SDEs are typically approximated numerically using schemes such as Euler–Maruyama on a discretized time grid. However, even with such approximations, traditional inference methods, such as MCMC, often fail to scale in settings with high-dimensional latent variables.

Variational inference is a more recent Bayesian inference method that transforms the integration problem into an optimization problem. The core idea is to use a family of variational distributions, parameterized by variational parameters, that are easy to sample from. The optimization problem involves finding the member of the variational family which best approximates the true posterior distribution of the latent variables. Several authors have applied variational inference to SDEs (Ryder et al., 2018; Ghosh et al., 2022;

Tzen and Raginsky, 2019), and to SSMs more generally (Archer et al., 2015; Becker et al., 2019; Ruhe and Forré, 2022; Krishnan et al., 2017; Revach et al., 2022).

Our research contributes to the SDE parameter inference literature by providing a IEVI method that accurately estimates the parameter posteriors of SDEs with high-dimensional latent variables, while being several orders of magnitude faster than standard MCMC methods. The remainder of this chapter outlines the contributions of this thesis and the structure of this document.

In Chapter 2, we investigate the convergence rates of the probabilistic solvers based on the Kalman algorithm for approximating the ODE solution. While probabilistic solvers using the forward Kalman filter to approximate the ODE solution have demonstrated convergence (Kersting et al., 2020b), the same cannot be said for the backward Kalman smoother. We bridge this gap by proving analogous convergence results for the backward pass and present numerical experiments that provide empirical support for our theory. Additionally, we provide empirical evidence that supports the theory of Rauch et al. (1965), demonstrating that the Kalman smoother estimates are more accurate than those of the Kalman filter alone. We implement a fast, automatically differentiable, and just-in-time (JIT) compiled probabilistic ode solver (rodeo) based on this Bayesian filtering approach using **JAX** in Python.

Chapter 2 also surveys several parameter inference methods based on the Bayesian filtering framework and introduces a novel variation of the MAnifold-constrained Gaussian process Inference (MAGI) method (Yang et al., 2021). Our variant introduces an additional modelling assumption to enable a more efficient algorithm. We implement this new method, along with several existing approaches, within the **rodeo** framework and evaluate their performance.

Chapter 3 introduces probabilistic solvers which can directly use noisy data to help guide the solution process. These solvers have shown empirical evidence of reducing parameter sensitivity more effectively than deterministic algorithms (Tronarp et al., 2022). However, sensitivity in the ODE model parameters remains an ongoing problem that has yet to be fully solved. We introduce a novel data-adaptive probabilistic ODE likelihood approximation (DALTON), which has several notable features. First, it is a computationally efficient method that adopts best practices from Krämer et al. (2022), enabling linear scaling in both solution discretization time points and ODE variables. Second, it is applicable to partially unobserved data and general non-Gaussian measurement errors, providing flexibility in the measurement model. Finally, DALTON can overcome deep local modes in the parameter likelihood function of chaotic ODE systems, a challenge that previous probabilistic and deterministic solvers have fallen short in addressing.

In Chapter 4, we focus on the parameter inference problem for SDEs. We present a information-efficient variational inference (IEVI) method that leverages advances in variational inference to convert the integration problem into a tractable optimization problem. IEVI is a fast, approximate Bayesian inference framework that effectively captures the forward–backward information propagation in SSMs (i.e., conditioning on both past and future observations), while scaling linearly with the number of latent variables. Benefiting from recent developments in neural networks and gradient-based optimization, our method can accurately estimate parameter posteriors for problems involving thousands of latent variables.

Chapter 2

Bayesian Filtering Probabilistic Solvers for Ordinary Differential Equations

2.1 Introduction

Probabilistic ODE solvers are designed to solve arbitrary-order multivariable systems which satisfy an initial value problem (IVP). For ease of presentation, we focus on first-order systems of the form

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(\boldsymbol{x}(t), t), \qquad \boldsymbol{x}(0) = \boldsymbol{v}, \qquad t \in [0, T], \tag{2.1}$$

where $\boldsymbol{x}(t) = \big(x_0(t), \ldots, x_{d-1}(t)\big)$ is d-dimensional and $\dot{\boldsymbol{x}}(t) = \frac{\mathrm{d}\boldsymbol{x}(t)}{\mathrm{d}t}$ is a shorthand notation for the derivative. Typically, $\boldsymbol{f} = \big(f_0, \ldots, f_{d-1}\big)$ are nonlinear functions with f_k representing a differential equation and $\boldsymbol{v} = \big(v_0, \ldots, v_{d-1}\big)$ are the initial values for $k = 0, \ldots d-1$.

Here, we adopt a zero-based indexing for vectors and matrices, i.e., $\boldsymbol{X} = (x^{(0)}, \ldots, x^{(p-1)}) \in \mathbb{R}^p$. For a matrix $\boldsymbol{P} \in \mathbb{R}^{m \times n}$ with index $0 \leq i \leq m-1$ and $0 \leq j \leq n-1$, we define $\boldsymbol{P}^{(i,:)}$ to denote the i-th row vector, $\boldsymbol{P}^{(:,j)}$ to be the j-th column vector and $\boldsymbol{P}^{(i,j)}$ to be the element in the i-th row and j-th column. The reason for this choice of notation is that the derivative for the IVP starts at the zeroth order, and indexing in this manner will be less confusing.

A family of probabilistic solvers (Tronarp et al., 2018; Krämer and Hennig, 2020; Kersting et al., 2020b; Krämer et al., 2022; Tronarp et al., 2022) solves the IVP in (2.1) using

the Bayesian nonlinear filtering paradigm of Tronarp et al. (2018). These solvers rely on the Kalman filtering and smoothing algorithms to achieve comparable speeds and accuracy to deterministic solvers, such as the Runge-Kutta method (Dormand and Prince, 1980), on a wide range of IVPs. As a result, this chapter focuses on probabilistic solvers which adopt the Bayesian filtering paradigm.

Theory on the convergence of probabilistic solvers using the Kalman filter has previously been established by Kersting et al. (2020b). However, the same cannot be said for solvers using the backward RTS smoother. In this chapter, our goal is to bridge this gap by establishing similar convergence rates for the backward pass. We substantiate our theory with empirical evidence from numerical studies. Furthermore, we present empirical results demonstrating that RTS smoother estimates are more accurate than those of the Kalman filter alone, aligning with the theory presented in Rauch et al. (1965).

Having established a theoretical foundation for Kalman-based probabilistic solvers and their convergence properties, we also address the problem of parameter inference, which requires repeatedly solving the ODE. We review existing parameter inference methods from the literature (Chkrebtii et al., 2016; Tronarp et al., 2022) and introduce a novel adaptation of the MAGI method of Yang et al. (2021). This adapted approach builds upon the Gaussian Markov process prior assumption of the probabilistic solvers, enabling more efficient parameter estimation.

To facilitate this, we develop a fast and accurate ODE solver based on the Kalman filter (Chkrebtii et al., 2016; Tronarp et al., 2018) and implement the parameter inference methods discussed later in this chapter within the Python library **rodeo**. Built using the **JAX** library (Bradbury et al., 2018), **rodeo** is written in automatic differentiable and JIT-compiled code, providing an application programming interface similar to **NumPy**.

The remainder of this chapter is organized as follows. Section 2.2 discusses how the SSM is used to represent the IVP, justifies the choice of a prior for the solution process, and introduces an approximation to the true model that enables efficient computation. Section 2.3 describes the Kalman filter and the RTS smoother along with their corresponding equations. Using these equations, we study the convergence results from the Kalman filter (Kersting et al., 2020b) and extend them to establish the same convergence rates for the RTS smoother. Section 2.4 presents an empirical study that corroborates these theoretical findings. Section 2.5 introduces the parameter inference problem, discussing various methods that leverage probabilistic solvers, including our novel adaptation of the MAGI method. Section 2.6 provides a numerical experiment comparing the methods. Finally, Section 2.7 summarizes key conclusions and outlines potential directions for future research.

2.2 State-Space Model Representing the Initial Value Problem

The Bayesian nonlinear filtering approach begins by assuming that $\boldsymbol{x}(t)$ and its first $p_k - 1$ derivatives denoted by $\boldsymbol{X}(t) = (\boldsymbol{X}_0(t), \ldots, \boldsymbol{X}_{d-1}(t))$ have a Gaussian Markov process prior where

$$\boldsymbol{X}_k(t) = \left(x_k(t), \dot{x}_k(t), \ldots, \frac{\mathrm{d}^{p_k-1}}{\mathrm{d}t^{p_k-1}} x_k(t) \right) \tag{2.2}$$

for $k = 0, \ldots, d-1$. The purpose of including higher-order derivatives will be made clear in Section 2.2.1. The restriction to a Gaussian and Markovian prior is to enable Kalman filtering and RTS smoothing to solve the ODE in linear time $O(N)$, with respect to the number of time discretization points (Tronarp et al., 2018; Schober et al., 2019). Let $t = t_0, \ldots, t_N$ where $t_n = n \cdot \Delta t$ and $\Delta t = T/N$ be the discretization time points for which the probabilistic solver gives an approximation to the solution. For clarity of notation, we define $\boldsymbol{x}_n = \boldsymbol{x}(t_n)$ and general indexing $\boldsymbol{x}_{m:n} = (\boldsymbol{x}_m, \ldots, \boldsymbol{x}_n)$ where $m \leq n$. Suppose $\boldsymbol{x}_{0:N}$ is the solution to (2.1), then we ought to have $\boldsymbol{Z}_n = \dot{\boldsymbol{x}}_n - \boldsymbol{f}(\boldsymbol{x}_n, t_n) = \boldsymbol{0}$. This information from the IVP described by $\boldsymbol{Z}_n$ can be used to update the Markov prior. Specifically, Tronarp et al. (2018) propose a SSM which represent $\boldsymbol{X}_n$ and $\boldsymbol{Z}_n$ by

$$\begin{aligned} \boldsymbol{X}_{n+1} \mid \boldsymbol{X}_n &\sim \mathrm{Normal}(\boldsymbol{Q}\boldsymbol{X}_n, \boldsymbol{R}) \\ \boldsymbol{Z}_{n+1} &\overset{\text{ind}}{\sim} \mathrm{Normal}(\dot{\boldsymbol{x}}_{n+1} - \boldsymbol{f}(\boldsymbol{x}_{n+1}, t_{n+1}), \boldsymbol{V}), \end{aligned} \tag{2.3}$$

where $\boldsymbol{x}_0 = (v_0, \ldots, v_{d-1})$ is the initial value, $\boldsymbol{Q} = \boldsymbol{Q}(\Delta t)$ and $\boldsymbol{R} = \boldsymbol{R}(\Delta t)$ are determined by the Gaussian Markov process prior. Then the stochastic solution to the IVP can be obtained by finding the posterior distribution

$$p(\boldsymbol{X}_{1:N} \mid \boldsymbol{Z}_{1:N} = \boldsymbol{0}) \tag{2.4}$$

to model (2.3).

2.2.1 Gaussian Markov Process Prior

For computational efficiency, it is common to assume independent Gaussian Markov process priors for $\boldsymbol{X}_k(t)$ (Krämer et al., 2022) where $k = 0, \ldots, d-1$. The goal is to enable matrix operations to be performed independently for each variable. Further, a simple prior to use

for each dimension k is the $p_k - 1$-times integrated Brownian motion (IBM) (Tronarp et al., 2018; Kersting et al., 2020b) given by

$$\frac{\mathrm{d}^{p_k-1}}{\mathrm{d}t^{p_k-1}} x_k(t) = \sigma_k B_k(t) \tag{2.5}$$

where σ_k is a tuning parameter and B_k is a one-dimensional Brownian motion.

Since model (2.3) involves the first derivative, we must have $p_k - 1 \geq 1$. For simplicity, we assume $p_k = 2$ for all $k = 0, \ldots, d-1$ throughout the remainder of this chapter. However, using a larger p_k results in a smoother solution process, which may improve approximation accuracy. Assuming this prior gives a p_k-dimensional continuous Gaussian Markov process on $\boldsymbol{X}_k(t)$ defined by

$$\boldsymbol{X}_k(t + \Delta t) \mid \boldsymbol{X}_k(t) \sim \mathrm{Normal}(\boldsymbol{Q}^{(k)} \boldsymbol{X}_k(t), \boldsymbol{R}^{(k)}), \tag{2.6}$$

where matrices $\boldsymbol{Q}^{(k)}, \boldsymbol{R}^{(k)} \in \mathbb{R}^{p \times p}$ are given by

$$Q_{ij}^{(k)} = \mathbb{1}_{i \leq j} \frac{(\Delta t)^{j-i}}{(j-i)!}, \qquad R_{ij}^{(k)} = \sigma_k^2 \frac{(\Delta t)^{2p_k - 1 - i - j}}{(2p_k - 1 - i - j)(p_k - 1 - i)!(p_k - 1 - j)!}. \tag{2.7}$$

Stacking $\boldsymbol{Q}^{(k)}$ and $\boldsymbol{R}^{(k)}$ in a block diagonal fashion gives the corresponding matrices in (2.3),

$$\boldsymbol{Q} = \begin{bmatrix} \boldsymbol{Q}^{(0)}_{p_0 \times p_0} & & \\ & \ddots & \\ & & \boldsymbol{Q}^{(d-1)}_{p_{d-1} \times p_{d-1}} \end{bmatrix}, \qquad \boldsymbol{R} = \begin{bmatrix} \boldsymbol{R}^{(0)}_{p_0 \times p_0} & & \\ & \ddots & \\ & & \boldsymbol{R}^{(d-1)}_{p_{d-1} \times p_{d-1}} \end{bmatrix}. \tag{2.8}$$

2.2.2 Model Interrogations

It is noted by Tronarp et al. (2021) that the mode of the posterior distribution (2.4) converges to the true ODE solution as $\Delta t \to 0$ and $\boldsymbol{V} \to 0$. Unfortunately, this posterior is typically intractable and cannot be sampled from directly due to the possible nonlinearity of $\boldsymbol{f}$ in (2.3). Of possible alternatives, MCMC (Yang et al., 2021) and particle filtering (Tronarp et al., 2018) have been studied closely. These methods provide accurate solutions at the cost of high computational complexity and can be especially slow for high-dimensional ODE problems.

A significantly faster but less accurate approach is to linearize (2.3), giving the surrogate model

$$\begin{aligned} \boldsymbol{X}_{n+1} \mid \boldsymbol{X}_n &\sim \mathrm{Normal}(\boldsymbol{Q}\boldsymbol{X}_n, \boldsymbol{R}) \\ \boldsymbol{Z}_{n+1} &\overset{\text{ind}}{\sim} \mathrm{Normal}(\dot{\boldsymbol{x}}_{n+1} + \boldsymbol{B}_{n+1}\boldsymbol{x}_{n+1} + \boldsymbol{a}_{n+1}, \boldsymbol{V}_{n+1}), \end{aligned} \tag{2.9}$$

where $\boldsymbol{a}_{n+1}, \boldsymbol{B}_{n+1}$ and $\boldsymbol{V}_{n+1}$ are computed via an online process known as *model interrogations* (Chkrebtii et al., 2016). We will explain how exactly this is done shortly. The main point is that the surrogate SSM in (2.9) is entirely linear and analytically tractable using the Kalman filter and the RTS smoother, allowing for linear time complexity in the number of discretization points. We denote the solution posterior to this surrogate model (2.9) by $p_{\text{lin}}(\boldsymbol{X}_{1:N} \mid \boldsymbol{Z}_{1:N} = \boldsymbol{0})$.

Next, we introduce four widely used model interrogation techniques from the literature. The first method, proposed by Schober et al. (2019), is a zeroth-order Taylor expansion given by

$$\boldsymbol{f}(\boldsymbol{x}_{n+1}, t_{n+1}) \approx \boldsymbol{f}(\boldsymbol{\mu}^{(0)}_{n+1|n}, t_{n+1}), \tag{2.10}$$

where $\boldsymbol{\mu}^{(0)}_{n+1|n} = E_{\text{lin}}[\boldsymbol{x}_{n+1} \mid \boldsymbol{Z}_{1:n}]$ is obtained from the predicted mean as calculated by the Kalman filter. Details on how this quantity is computed in an online manner are provided in Section 2.3.

The second method, proposed by Tronarp et al. (2018), extends this to a first-order Taylor expansion, which is given by

$$\boldsymbol{f}(\boldsymbol{x}_{n+1}, t_{n+1}) \approx \boldsymbol{f}(\boldsymbol{\mu}^{(0)}_{n+1|n}, t_{n+1}) + \boldsymbol{J}_f(\boldsymbol{\mu}^{(0)}_{n+1|n}, t_{n+1})(\boldsymbol{x}_{n+1} - \boldsymbol{\mu}^{(0)}_{n+1|n}), \tag{2.11}$$

where $\boldsymbol{\mu}^{(0)}_{n+1|n} = E_{\text{lin}}[\boldsymbol{x}_{n+1} \mid \boldsymbol{Z}_{1:n}]$ is obtained from the predicted mean as calculated by the Kalman filter and $\boldsymbol{J}_f$ is the Jacobian of $\boldsymbol{f}(\boldsymbol{x}, t)$ with respect to $\boldsymbol{x}$. However, a key limitation of this method is that the Jacobian matrix $\boldsymbol{J}_f$ is not necessarily diagonal, as one of the functions $\boldsymbol{f}_k$ could depend on multiple variables for $k = 0, \ldots, d-1$. As a result, this method breaks the independence structure of $\boldsymbol{X}_k$ as in the case of the method of Schober et al. (2019), and matrix operations can no longer be performed independently across dimensions. This dramatically increases the computational complexity from $\mathcal{O}(d)$ to $\mathcal{O}(d^3)$ (Krämer et al., 2022).

This leads to the third method proposed by Krämer et al. (2022), which is a slight modification of the second method. Instead of using the full Jacobian, Krämer et al. (2022) approximate it using only the diagonal components; we denote this approximation as $\boldsymbol{J}^*_f$. This ensures that each variable is entirely independent in the approximate SSM (2.9), and computations can be done independently along the dimensions (Krämer et al., 2022). Moreover, Krämer et al. (2022) demonstrate that this diagonal approximation achieves similar accuracy to the full Jacobian method while significantly reducing computation time in empirical ODE examples. Compared to the method of Schober et al. (2019), it is only slightly slower, while providing a significant improvement in the accuracy of the ODE approximation (Krämer et al., 2022).

Up until now, all model interrogation techniques have assumed zero variance in $\boldsymbol{Z}_{n+1}$ when approximating $\boldsymbol{f}(\boldsymbol{x}_{n+1}, t_{n+1})$ with linearization, i.e., $\boldsymbol{V}_{n+1} = \boldsymbol{0}$ (Schober et al., 2019). The fourth method, introduced by Chkrebtii et al. (2016) takes a different approach: rather than linearizing $\boldsymbol{f}(\boldsymbol{x}_{n+1}, t_{n+1})$ using $\boldsymbol{f}(\boldsymbol{\mu}^{(0)}_{n+1|n}, t_{n+1})$, they approximate $E[\boldsymbol{f}(\boldsymbol{x}_{n+1}, t_{n+1}) \mid \boldsymbol{Z}_{1:n} = \boldsymbol{0}]$ via Monte Carlo sampling (Schober et al., 2019) from a normal distribution with mean $\boldsymbol{\mu}^{(0)}_{n+1|n}$ and variance $\boldsymbol{\Sigma}^{(0,0)}_{n+1|n}$. Furthermore, Chkrebtii et al. (2016) note that the noise-free model interrogations with $\boldsymbol{V}_{n+1} = 0$ degrades solver performance and instead propose setting $\boldsymbol{V}_{n+1} = \boldsymbol{\Sigma}^{(1,1)}_{n+1|n}$. Details on computing the mean and variance in an online manner during Kalman filtering are provided in Section 2.3. Table 2.1 summarizes the model interrogation methods found in the literature.

Table 2.1: Various model interrogation methods.

Reference	$\boldsymbol{a}_{n+1}$	$\boldsymbol{B}_{n+1}$	$\boldsymbol{V}_{n+1}$
Chkrebtii et al. (2016)	$-\boldsymbol{f}(\boldsymbol{x}^{\star}_{n+1}, t_{n+1}) : \boldsymbol{x}^{\star}_{n+1} \sim \text{Normal}(\boldsymbol{\mu}^{(0)}_{n+1|n}, \boldsymbol{\Sigma}^{(0,0)}_{n+1|n})$	$\boldsymbol{0}$	$\boldsymbol{\Sigma}^{(1,1)}_{n+1|n}$
Schober et al. (2019)	$-\boldsymbol{f}(\boldsymbol{\mu}^{(0)}_{n+1|n}, t_{n+1})$	$\boldsymbol{0}$	$\boldsymbol{0}$
Tronarp et al. (2018)	$-\boldsymbol{f}(\boldsymbol{\mu}^{(0)}_{n+1|n}, t_{n+1}) + \boldsymbol{J}_f(\boldsymbol{\mu}^{(0)}_{n+1|n}, t_{n+1})\boldsymbol{\mu}^{(0)}_{n+1|n}$	$-\boldsymbol{J}_f(\boldsymbol{\mu}^{(0)}_{n+1|n}, t_{n+1})$	$\boldsymbol{0}$
Krämer et al. (2022)	$-\boldsymbol{f}(\boldsymbol{\mu}^{(0)}_{n+1|n}, t_{n+1}) + \boldsymbol{J}^{*}_f(\boldsymbol{\mu}^{(0)}_{n+1|n}, t_{n+1})\boldsymbol{\mu}^{(0)}_{n+1|n}$	$-\boldsymbol{J}^{*}_f(\boldsymbol{\mu}^{(0)}_{n+1|n}, t_{n+1})$	$\boldsymbol{0}$

Currently, theoretical results (Kersting et al., 2020b) for the convergence rates of the surrogate model (2.9) are only available for the zeroth-order linearization, as proposed by Schober et al. (2019).

2.3 Kalman Filtering and Smoothing

The Kalman filter estimates the distribution of the latent variable conditioned on the measurements seen thus far. In this case, the latent variable is $\boldsymbol{X}_{0:N}$ and the "measurements" are $\boldsymbol{Z}_{1:N} = \boldsymbol{0}$, as given by the ODEs. It is a recursive algorithm that estimates the distribution $p_{\text{lin}}(\boldsymbol{X}_n \mid \boldsymbol{Z}_{1:n} = \boldsymbol{0})$ at each step for $n = 1, \ldots, N$. Since both latent and measurements are Gaussian, the conditional distribution is also Gaussian. Therefore, the algorithm only needs to keep track of the mean and variance, resulting in space and time complexity that scale linearly with N. For these estimates, we use the general notation

$$\begin{aligned} \boldsymbol{\mu}_{n|m} &= E_{\text{lin}}[\boldsymbol{X}_n \mid \boldsymbol{Z}_{1:m} = \boldsymbol{0}] \\ \boldsymbol{\Sigma}_{n|m} &= \text{var}_{\text{lin}}[\boldsymbol{X}_n \mid \boldsymbol{Z}_{1:m} = \boldsymbol{0}]. \end{aligned} \tag{2.12}$$

To avoid the confusion of added indices, we shall assume that $d = 1$ for the remainder of this chapter, i.e., the IVP (2.1) is a univariate system. As noted by Kersting et al. (2020b), the generalization to $d > 1$ essentially amounts to applying the proofs and equations in the rest of this section to each dimension independently (Kersting et al., 2020b). This is due to the independent nature we assumed for the Gaussian Markov process prior of $\boldsymbol{X}(t)$ in (2.5) and the interrogation method of Schober et al. (2019).

The Kalman filter consists of two distinct phases at each time step (Kalman, 1960): prediction and update. In the prediction phase, the current measurement, $\boldsymbol{Z}_n$, is assumed to be unobserved. Therefore, the predicted mean and variance are given by $\boldsymbol{\mu}_{n|n-1}$ and $\boldsymbol{\Sigma}_{n|n-1}$, respectively, for $n = 1, \ldots, N$. During the update phase, we assume the current measurement is observed and update our prediction estimates with this new information. The updated mean and variance are denoted by $\boldsymbol{\mu}_{n|n}$ and $\boldsymbol{\Sigma}_{n|n}$, respectively, for $n = 1, \ldots, N$.

The Kalman filter is a recursive algorithm in which the current state depends only on the most recent previous state. This is due to the nature of the Markov prior we assumed. Suppose we are at step $n + 1$, where we have the updated mean and variance from the previous step n. The prediction phase uses all the information known up to this time to predict the current state, which can be computed by (Kalman, 1960)

$$\begin{aligned} \boldsymbol{\mu}_{n+1|n} &= \boldsymbol{Q}\boldsymbol{\mu}_{n|n} \\ \boldsymbol{\Sigma}_{n+1|n} &= \boldsymbol{Q}\boldsymbol{\Sigma}_{n|n}\boldsymbol{Q}^\top + \boldsymbol{R}. \end{aligned} \tag{2.13}$$

The next stage of the Kalman filter is the update phase, in which the updated mean and variance are given by

$$\begin{aligned} \boldsymbol{\mu}_{n+1|n+1} &= \boldsymbol{\mu}_{n+1|n} + \boldsymbol{\beta}_{n+1}\boldsymbol{r}_{n+1} \\ \boldsymbol{\Sigma}_{n+1|n+1} &= \boldsymbol{\Sigma}_{n+1|n} - \boldsymbol{\beta}_{n+1}\boldsymbol{W}_{n+1}\boldsymbol{\Sigma}_{n+1|n}, \end{aligned} \tag{2.14}$$

where $\boldsymbol{\beta}_{n+1}$ is the Kalman gain, $\boldsymbol{r}_{n+1}$ is the so-called innovation and $\boldsymbol{W}_{n+1} = (\boldsymbol{W}^{(1)} + \boldsymbol{B}_{n+1}\boldsymbol{W}^{(0)})$. Here, the matrices $\boldsymbol{W}^{(i)} = \begin{bmatrix} 1-i & i \end{bmatrix}$ for $i = 0, 1$ are selection matrices used to obtain the zeroth and the first derivative, respectively. In other words, the mean of the measurement model in (2.9) can be written as

$$\begin{aligned} \dot{\boldsymbol{x}}_{n+1} + \boldsymbol{B}_{n+1}\boldsymbol{x}_{n+1} + \boldsymbol{a}_{n+1} &= \boldsymbol{W}^{(1)}\boldsymbol{X}_{n+1} + \boldsymbol{B}_{n+1}\boldsymbol{W}^{(0)}\boldsymbol{X}_{n+1} + \boldsymbol{a}_{n+1} \\ &= (\boldsymbol{W}^{(1)} + \boldsymbol{B}_{n+1}\boldsymbol{W}^{(0)})\boldsymbol{X}_{n+1} + \boldsymbol{a}_{n+1}. \end{aligned} \tag{2.15}$$

The Kalman gain can be thought of as the weight assigned to the measurements relative

to the predicted mean and is given by

$$\begin{aligned}\boldsymbol{\beta}_{n+1} &= (\beta_{n+1}^{(0)}, \beta_{n+1}^{(1)})^\top \\ &= \boldsymbol{\Sigma}_{n+1|n}\boldsymbol{W}_{n+1}^\top[\boldsymbol{W}_{n+1}\boldsymbol{\Sigma}_{n+1|n}\boldsymbol{W}_{n+1}^\top + \boldsymbol{V}_{n+1}]^{-1}.\end{aligned} \tag{2.16}$$

The innovation is defined as the difference between the actual measurement and the optimal forecast of that value (Kalman, 1960). Under the surrogate model in (2.9) and the zeroth-order linearization (Schober et al., 2019), this is given by

$$\begin{aligned}\boldsymbol{r}_{n+1} &:= \boldsymbol{Z}_{n+1} - \boldsymbol{\mu}_{n+1|n}^{(1)} - \boldsymbol{B}_{n+1}\boldsymbol{\mu}_{n+1|n}^{(0)} - \boldsymbol{a}_{n+1} \\ &= -(\boldsymbol{W}^{(1)} + \boldsymbol{B}_{n+1}\boldsymbol{W}^{(0)})\boldsymbol{\mu}_{n+1|n} - \boldsymbol{a}_{n+1} \\ &= -\boldsymbol{W}_{n+1}\boldsymbol{\mu}_{n+1|n} - \boldsymbol{a}_{n+1},\end{aligned} \tag{2.17}$$

where the measurements are known to be $\boldsymbol{Z}_{1:N} = \boldsymbol{0}$.

This concludes one step of the Kalman filter. Using the same recursion, we can find the corresponding estimates for the mean $\boldsymbol{\mu}_{n|n}$ and variance $\boldsymbol{\Sigma}_{n|n}$ for $n = 1, \ldots, N$. To initialize the recursion, we use $\boldsymbol{\mu}_{0|0} = (v, f(v, t_0))$ and $\boldsymbol{\Sigma}_{0|0} = \boldsymbol{0}$, since we know the exact initial value with zero uncertainty.

The Kalman filter computes $p_{\text{lin}}(\boldsymbol{X}_n \mid \boldsymbol{Z}_{1:n} = \boldsymbol{0})$ for $n = 1, \ldots, N$. However, our goal is to compute $p_{\text{lin}}(\boldsymbol{X}_{1:N} \mid \boldsymbol{Z}_{1:N} = \boldsymbol{0})$. The subtlety here is that the solution posterior we are after incorporates all the measurements for each time step, rather than only the previous ones. More precisely, instead of $p_{\text{lin}}(\boldsymbol{X}_n \mid \boldsymbol{Z}_{1:n} = \boldsymbol{0})$, we aim to find $p_{\text{lin}}(\boldsymbol{X}_n \mid \boldsymbol{Z}_{1:N} = \boldsymbol{0})$ for $n = 1, \ldots, N$. To do this, we employ the RTS smoothing algorithm.

Similar to the Kalman filter, the RTS smoother is also a recursive algorithm. It begins by assuming the Kalman filter has been completed up to the last step, with all predicted and updated estimates stored in memory. Then, starting at the end of the Kalman filter, which is at time point t_N, the algorithm works backward in time towards time point t_0. At time point t_n, we denote the smoothed mean and variance as $\boldsymbol{\mu}_{n|N}$ and $\boldsymbol{\Sigma}_{n|N}$, respectively.

The RTS smoother uses the predicted and updated estimates at time t_n to compute the corresponding smoothed mean and variance using the following equations (Rauch et al., 1965)

$$\begin{aligned}\boldsymbol{\mu}_{n|N} &= \boldsymbol{\mu}_{n|n} + \boldsymbol{A}_n(\boldsymbol{\mu}_{n+1|N} - \boldsymbol{\mu}_{n+1|n}) \\ \boldsymbol{\Sigma}_{n|N} &= \boldsymbol{\Sigma}_{n|n} + \boldsymbol{A}_n(\boldsymbol{\Sigma}_{n+1|N} - \boldsymbol{\Sigma}_{n+1|n})\boldsymbol{A}_n^\top\end{aligned} \tag{2.18}$$

where

$$\boldsymbol{A}_n = \boldsymbol{\Sigma}_{n|n}\boldsymbol{Q}^\top\boldsymbol{\Sigma}_{n+1|n}^{-1}. \tag{2.19}$$

To initialize the RTS smoother, the smoothed estimates at time t_N are set to the updated estimates, as they already condition on all measurements. At time point t_0, where the initial value is given, we set $\boldsymbol{\mu}_{0|N} = (v, f(v, t_0))$ and $\boldsymbol{\Sigma}_{0|N} = \mathbf{0}$.

To summarize the ODE solver using the Kalman recursions described above, we outline the main steps in Algorithm 2.1. Additionally, we introduce the `kalman_ode_sim()` method, which generates a sample solution from the posterior $\boldsymbol{X}_{0:N} \sim p_{\text{lin}}(\boldsymbol{X}_{1:N} \mid \boldsymbol{Z}_{1:N} = \mathbf{0})$ (Doucet and Johansen, 2009), accounting for scenarios where a sample is preferred over the mean. The related Kalman functions within Algorithm 2.1, as presented in (2.13), (2.14), and (2.18), are available in Algorithm A.1 in Appendix A.2 for reference.

2.3.1 Convergence of the Kalman Filter Estimates

In this section, we present the main results of Kersting et al. (2020b), which establish the convergence rates of the Kalman filter estimates under model (2.9), assuming the interrogation method of Schober et al. (2019). To do so, we first introduce the following necessary assumptions.

Assumption 2.1. *The function $f : \mathbb{R} \times [0, T] \to \mathbb{R}$ is globally Lipschitz in x, continuous in t, and its partial derivative with respect to x is uniformly bounded and globally Lipschitz, i.e., there exists some $L > 0$ such that*

$$|f(x,t) - f(y,t)| \leq L|x - y|, \qquad \left|\frac{\partial f(x,t)}{\partial x} - \frac{\partial f(y,t)}{\partial x}\right| \leq L|x - y|, \qquad \left|\frac{\partial f(x,t)}{\partial x}\right| \leq L,$$

for all $x, y \in \mathbb{R}$ over the interval $[0, T]$.

Under Assumption 2.1, the Picard–Lindelöf theorem ensures that there exists a unique solution to the IVP over the interval $[0, T]$.

Assumption 2.2. *The initial mean is the exact initialization $\boldsymbol{\mu}_{0|0} = (v, f(v, 0))$ and the initial variance matrix is $\boldsymbol{\Sigma}_{0|0} = \mathbf{0}$.*

Assumption 2.2 restricts the initial value and its derivative of the IVP to be correct with zero uncertainty ensuring that the probabilistic solution starts at the true value.

Assumption 2.3. *The tuning parameter in (2.9) is assumed to be $\boldsymbol{V}_n = \mathbf{0}$.*

These assumptions allow for the establishment of the following theorems.

Algorithm 2.1 The Kalman filter and smoother ODE solver contains three main components.

- `kalman_ode_forward` computes the Kalman filter estimates, $\boldsymbol{\mu}_{n|n}$ and $\boldsymbol{\Sigma}_{n|n}$ for $n = 0, \ldots, N$.
- `kalman_ode_mv` computes the Kalman smoother estimates, $\boldsymbol{\mu}_{n|N}$ and $\boldsymbol{\Sigma}_{n|N}$
- `kalman_ode_sim` generates a sample solution from the posterior $p_{\text{lin}}(\boldsymbol{X}_{1:N} \mid \boldsymbol{Z}_{1:N} = \boldsymbol{0})$.

---

procedure `kalman_ode_forward`$(\boldsymbol{f}(\boldsymbol{x}, t), \boldsymbol{v}, \boldsymbol{Q}, \boldsymbol{R})$
  $\boldsymbol{\mu}_{0|0}, \Sigma_{0|0} \leftarrow \boldsymbol{v}, \boldsymbol{0}$
  $\boldsymbol{W}^{(1)} \leftarrow \begin{bmatrix} 0 & 1 \end{bmatrix}, \qquad \boldsymbol{W}^{(0)} \leftarrow \begin{bmatrix} 1 & 0 \end{bmatrix}$
  $\boldsymbol{Z}_{1:N} \leftarrow \boldsymbol{0}$
  for $n = 1 : N$ **do**
    $\boldsymbol{\mu}_{n|n-1}, \boldsymbol{\Sigma}_{n|n-1} \leftarrow$ `kalman_predict`$(\boldsymbol{\mu}_{n-1|n-1}, \boldsymbol{\Sigma}_{n-1|n-1}, \boldsymbol{0}, \boldsymbol{Q}, \boldsymbol{R})$
    $\boldsymbol{a}_n, \boldsymbol{B}_n, \boldsymbol{V}_n \leftarrow$ `interrogate`$(\boldsymbol{\mu}_{n|n-1}, \boldsymbol{\Sigma}_{n|n-1}, \boldsymbol{f}(\boldsymbol{x}, t))$ ▷ See Table 2.1
    $\boldsymbol{\mu}_{n|n}, \boldsymbol{\Sigma}_{n|n} \leftarrow$ `kalman_update`$(\boldsymbol{\mu}_{n|n-1}, \boldsymbol{\Sigma}_{n|n-1}, \boldsymbol{Z}_n, \boldsymbol{a}_n, \boldsymbol{W}^{(1)} + \boldsymbol{B}_n \boldsymbol{W}^{(0)}, \boldsymbol{Z}_n)$
  return $\boldsymbol{\mu}_{0|0}, \ldots, \boldsymbol{\mu}_{N|N}, \boldsymbol{\Sigma}_{0|0}, \ldots, \boldsymbol{\Sigma}_{N|N}$

procedure `kalman_ode_mv`$(\boldsymbol{f}(\boldsymbol{x}, t), \boldsymbol{v}, \boldsymbol{Q}, \boldsymbol{R})$
  $\boldsymbol{\mu}_{0|0}, \ldots, \boldsymbol{\mu}_{N|N}, \boldsymbol{\Sigma}_{0|0}, \ldots, \boldsymbol{\Sigma}_{N|N} \leftarrow$ `kalman_ode_forward`$(\boldsymbol{f}(\boldsymbol{x}, t), \boldsymbol{v}, \boldsymbol{Q}, \boldsymbol{R})$
  for $n = N - 1 : 0$ **do**
    $\boldsymbol{\mu}_{n|N}, \boldsymbol{\Sigma}_{n|N} \leftarrow$ `kalman_smooth`$(\boldsymbol{\mu}_{n+1|N}, \boldsymbol{\Sigma}_{n+1|N}, \boldsymbol{\mu}_{n|n}, \boldsymbol{\Sigma}_{n|n}, \boldsymbol{\mu}_{n+1|n}, \boldsymbol{\Sigma}_{n+1|n}, \boldsymbol{Q})$
  return $\boldsymbol{\mu}_{0:N|N}, \boldsymbol{\Sigma}_{0:N|N}$

procedure `kalman_ode_sim`$(\boldsymbol{f}(\boldsymbol{x}, t), \boldsymbol{v}, \boldsymbol{Q}, \boldsymbol{R})$
  $\boldsymbol{\mu}_{0|0}, \ldots, \boldsymbol{\mu}_{N|N}, \boldsymbol{\Sigma}_{0|0}, \ldots, \boldsymbol{\Sigma}_{N|N} \leftarrow$ `kalman_ode_forward`$(\boldsymbol{f}(\boldsymbol{x}, t), \boldsymbol{v}, \boldsymbol{Q}, \boldsymbol{R})$
  $\boldsymbol{X}_N \sim \text{Normal}(\boldsymbol{\mu}_{N|N}, \boldsymbol{\Sigma}_{N|N})$
  for $n = N - 1 : 0$ **do**
    $\boldsymbol{\mu}_n, \boldsymbol{\Sigma}_n \leftarrow$ `kalman_sample`$(\boldsymbol{x}_{n+1}, \boldsymbol{\mu}_{n|n}, \boldsymbol{\Sigma}_{n|n}, \boldsymbol{\mu}_{n+1|n}, \boldsymbol{\Sigma}_{n+1|n}, \boldsymbol{Q})$
    $\boldsymbol{X}_n \sim \text{Normal}(\boldsymbol{\mu}_n, \boldsymbol{\Sigma}_n)$
  return $\boldsymbol{X}_{0:N}$

Theorem 2.1. *Under Assumptions 2.1 to 2.3 and for all sufficiently small step size $\Delta t > 0$,*

$$\max_{0 \le n \le N} |\boldsymbol{\mu}_{n|n}^{(0)} - x_n| = O(\Delta t) \tag{2.20}$$

where $\boldsymbol{\mu}_{n|n}^{(0)} = E_{lin}[x_n \mid \boldsymbol{Z}_{1:n}]$ *is the zeroth index of the Kalman updated mean and* x_n *represents the true ODE solution at time* t_n*.*

Theorem 2.1 states that the mean of the Kalman filter under the surrogate model (2.9) converges to the exact ODE solution as $\Delta t \to 0$ and $\boldsymbol{V}_n = \mathbf{0}$. This theorem, by itself, is not very useful unless the variance $\boldsymbol{\Sigma}_{n|n}^{(0)}$ also converges to $\mathbf{0}$, which is established by the following.

Theorem 2.2. *Under Assumptions 2.1 to 2.3 and for all sufficiently small step size* $\Delta t > 0$*,*

$$\max_{0 \le n \le N} \boldsymbol{\Sigma}_{n|n}^{(0,0)} = O((\Delta t)^2) \tag{2.21}$$

where $\boldsymbol{\Sigma}_{n|n}^{(0,0)} = \mathrm{var}_{lin}[x_n \mid \boldsymbol{Z}_{1:n}]$ *is the* $0, 0$*-th element of the Kalman updated variance.*

Theorem 2.1 and 2.2 establish a desired property commonly found in deterministic ODE solvers, which is that as computational complexity increases, the approximation becomes more accurate.

2.3.2 Convergence of the RTS Smoother Estimates

In this section, we present two theorems which establish that the RTS smoother estimates have the same order of convergence as the corresponding Kalman filter estimates, assuming the interrogation method of Schober et al. (2019). The RTS estimates are generally favoured over the corresponding Kalman filter estimates (Rauch et al., 1965) because the former use all measurements at every time point, whereas the latter only use previous measurements. The intuition behind this preference is that knowing how the ODE behaves in the future could potentially improve current estimates of the solution process. The technical details of the lemmas and proofs leading up to the following theorems are provided in Appendix A.

Theorem 2.3. *Under Assumptions 2.1 to 2.3 and for all sufficiently small step size* $\Delta t > 0$*,*

$$\max_{0 \le n \le N} |\boldsymbol{\mu}_{n|N}^{(0)} - x_n| = O(\Delta t) \tag{2.22}$$

where $\boldsymbol{\mu}_{n|N}^{(0)} = E_{lin}[x_n \mid \boldsymbol{Z}_{1:N}]$ *is the* 0*-th index of the RTS smoothed mean and* x_n *is the true ODE solution.*

Theorem 2.4. *Under Assumptions 2.1 to 2.3, for all sufficiently small step size* $\Delta t > 0$*,*

$$\boldsymbol{\Sigma}_{n|N} = \boldsymbol{\Sigma}_{n|n} \tag{2.23}$$

where $\boldsymbol{\Sigma}_{n|N}$ *is the RTS smoothed variance and* $\boldsymbol{\Sigma}_{n|n}$ *is the Kalman updated variance.*

2.4 Numerical Experiments on Convergence Rates

In this section, we empirically check the convergence rates of Theorems 2.3 and 2.4. We achieve this through work precision diagrams, which compare accuracy to the number of discretization points, N. To evaluate accuracy, we use the maximum absolute error (MAE) as our evaluation metric, as defined in Theorem 2.3,

$$\mathrm{MAE}(\boldsymbol{\mu}_{0:N}) = \max_{0 \le n \le N} |\boldsymbol{\mu}_n^{(0)} - x_n|, \tag{2.24}$$

where $\boldsymbol{\mu}_{0:N}$ is either the Kalman updated mean or the RTS smoothed mean, and $x_{0:N}$ is the true solution. In contrast, we evaluate Theorem 2.4 using the maximum error in variance (MEV) as our evaluation metric, defined as

$$\mathrm{MEV}(\boldsymbol{\Sigma}_{0:N}) = \max_{0 \le n \le N} \boldsymbol{\Sigma}_n^{(0,0)}, \tag{2.25}$$

where $\boldsymbol{\Sigma}_{0:N}$ is either the Kalman updated variance or the RTS smoothed variance.

2.4.1 FitzHugh-Nagumo Model

The FitzHugh-Nagumo (FN) model (Sherwood, 2013) is a two-dimensional ODE on $\boldsymbol{x}(t) = \big(V(t), R(t)\big)$, in which $V(t)$ describes the evolution of the neuronal membrane voltage and $R(t)$ describes the activation and deactivation of neuronal channels. The FN-IVP is given by

$$\dot{V}(t) = c\Big(V(t) - \frac{V(t)^3}{3} + R(t)\Big), \quad \dot{R}(t) = -\frac{V(t) - a + bR(t)}{c}, \tag{2.26}$$

with initial values $\boldsymbol{x}(0) = (1, 0)$, parameters $(a, b, c) = (0.08, 0.07, 1.2)$ and $t \in [0, 10]$. Under these settings, no analytical closed-form solution exists. Thus, we use the solution from a highly accurate deterministic solver (Hindmarsh, 1992) as a proxy for the true ODE solution in Figure 2.1.

We conduct two experiments: one assumes an independent 1-time IBM prior, and the other an independent 2-times IBM prior (corresponding to $p = 2, 3$). We use scale parameters $\sigma_0 = \sigma_1 = 1$ for both experiments. Although Theorem 2.3 and 2.4 only apply to the 1-time IBM prior, our empirical results show similar convergence rates for the 2-time IBM prior as well. For the 2-times IBM prior, the second derivative is required in the initialization. This can be found by taking the derivative of $\dot{V}(t)$ and $\dot{R}(t)$. Another possible initialization is simply setting the second derivative to zero, i.e., $\boldsymbol{X}_0 = (\boldsymbol{V}_0, \boldsymbol{R}_0) = ([1, 0.8, 0], [0, -2.3/3, 0])$. Although this is not the exact initialization, our experiments

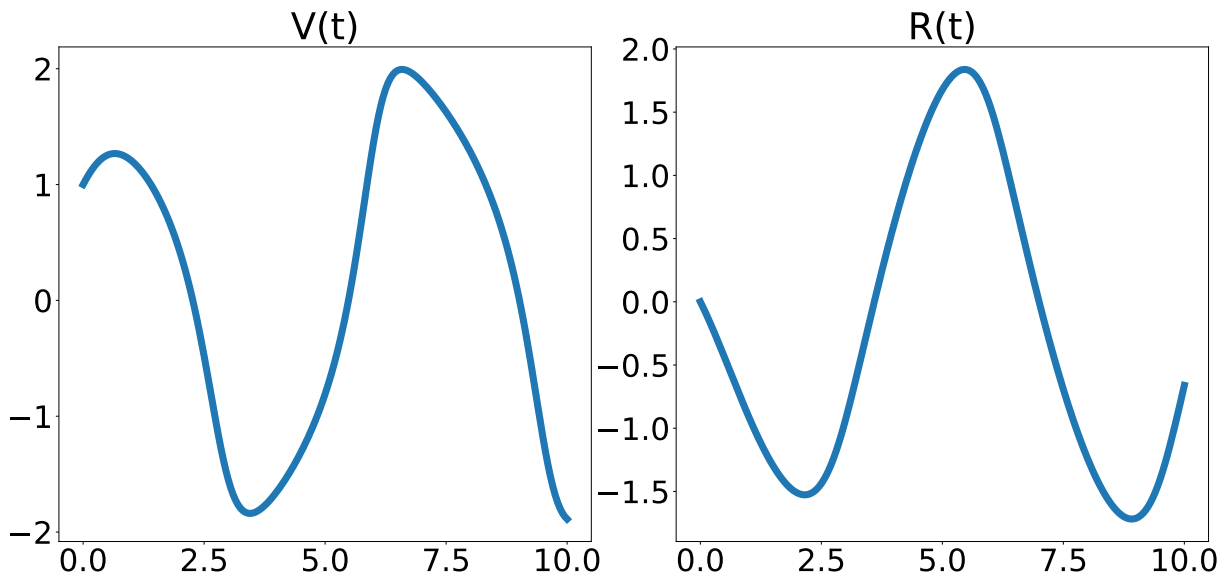

Figure 2.1: The true solution of the FN model.

showed that it has little effect on the solution process. This is likely due to the fact that the second derivative is not actually used in the FN model, and measurements only depend on the first derivative. This is rather convenient because it can be pretty tedious to compute the extra order of derivatives required in the initial value.

Figure 2.2 shows the convergence rates for the Kalman updated mean and the RTS smoothed mean at various discretization points N. For comparison, the lines corresponding to $h(N) = (10/N)^i = \Delta t^i$ for $i = 1, 2, 3$ are also displayed. In the case of the 2-times IBM prior, the convergence seems to be better than $O((\Delta t)^2)$ but worse than $O((\Delta t)^3)$. On the other hand, the 1-time IBM prior exhibits convergence rates better than $O(\Delta t)$ but worse than $O((\Delta t)^2)$, providing empirical evidence of Theorem 2.3. It appears that having a higher-order IBM prior improves the accuracy of the ODE estimate, but comes at the cost of increased computational complexity. Furthermore, the smoothed estimate is slightly more accurate than the updated estimate, supporting the theory presented in Rauch et al. (1965).

Figure 2.3 displays the convergence rates for the Kalman updated variance and the RTS smoothed variance. The two variances are exactly equal, complying with Theorem 2.4. Moreover, both variances have convergence rates of less than $O((\Delta t)^2)$ but more than $O((\Delta t)^3)$. Similar to the mean, adding an extra derivative in the prior results in a lower

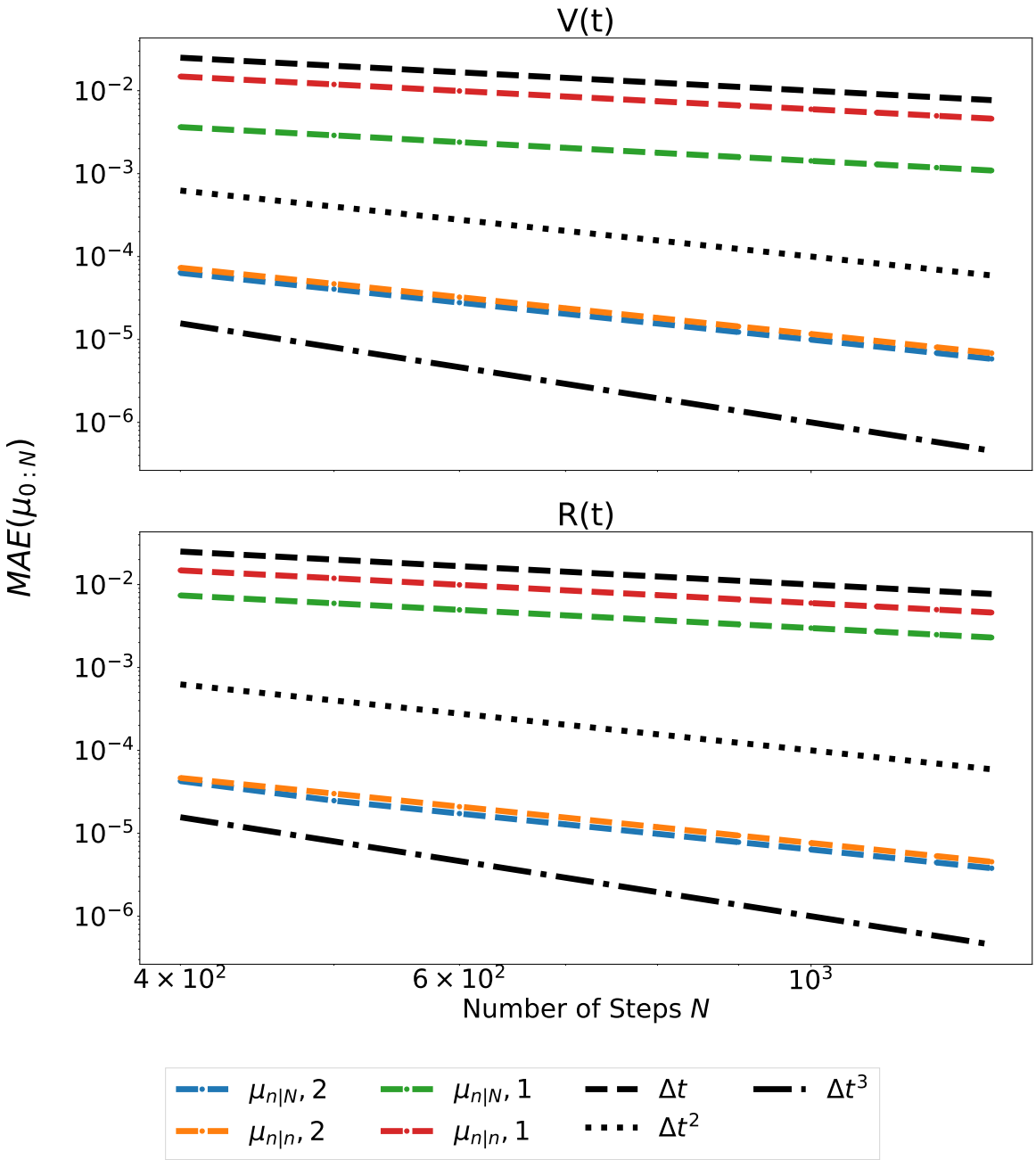

Figure 2.2: Convergence rates for the Kalman updated mean and the RTS smoothed mean using independent $\{1, 2\}$-times IBM prior.

order of convergence. In this case, the variance has a convergence rate lower than $O((\Delta t)^4)$.

2.5 Parameter Inference for Gaussian Filtering ODE Solvers

Having established the convergence rates of the probabilistic solvers introduced by Tronarp et al. (2018), we now shift our focus to the parameter inference problem for ODEs. The parameter-dependent extension of the IVP (2.1) is of the form

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{x}(t), t), \qquad \boldsymbol{x}(0) = \boldsymbol{v}_{\boldsymbol{\theta}}, \qquad t \in [0, T]. \tag{2.27}$$

In parameter learning problems, the solution process $\boldsymbol{x}(t)$ is not directly observed. Instead, noisy observations $\boldsymbol{Y}_{0:M} = (\boldsymbol{Y}_0, \ldots, \boldsymbol{Y}_M)$ are reported at discrete time points $t = t'_0, \ldots, t'_M$ under the observation model

$$\boldsymbol{Y}_i \overset{\text{ind}}{\sim} p(\boldsymbol{Y}_i \mid \boldsymbol{X}(t'_i), \boldsymbol{\phi}). \tag{2.28}$$

The goal is to estimate the model and noise parameters, $(\boldsymbol{\theta}, \boldsymbol{\phi})$, which determine the solution process $\boldsymbol{x}(t)$ in (2.27) from the observations.

To do this, we first need an ODE solver capable of computing the solution for any given $\boldsymbol{\theta}$. Under the Bayesian filtering paradigm of Tronarp et al. (2018), the updated surrogate model (2.9) in the context of parameter inference is given by

$$\begin{aligned} \boldsymbol{X}_{n+1} \mid \boldsymbol{X}_n &\sim \text{Normal}(\boldsymbol{Q}_{\boldsymbol{\eta}} \boldsymbol{X}_n, \boldsymbol{R}_{\boldsymbol{\eta}}) \\ \boldsymbol{Z}_{n+1} &\overset{\text{ind}}{\sim} \text{Normal}(\dot{\boldsymbol{x}}_{n+1} + \boldsymbol{B}_{n+1} \boldsymbol{x}_{n+1} + \boldsymbol{a}_{n+1}, \boldsymbol{V}_{n+1}), \end{aligned} \tag{2.29}$$

where the Gaussian Markov process prior depends on the tuning parameter $\boldsymbol{\eta}$.

A probabilistic ODE solver approximates the solution posterior, $p(\boldsymbol{X}_{1:N} \mid \boldsymbol{Z}_{1:N} = \boldsymbol{0}, \boldsymbol{\theta}, \boldsymbol{\eta})$, at the discretization time points $t = t_0, \ldots, t_N$ which could possibly differ from the observation time grid. To facilitate this, we define the mapping $n(\cdot)$ such that $t_{n(i)} = t'_i$ and assume that the time grid $\{t'_0, \ldots, t'_M\} \subseteq \{t_0, \ldots, t_N\}$. In other words, we assume that there is no time point where data is observed but no solution is approximated. Parameter inference is carried out with the likelihood function for the model, noise, and tuning parameters $\boldsymbol{\Theta} = (\boldsymbol{\theta}, \boldsymbol{\phi}, \boldsymbol{\eta})$, which can be formulated as

$$\begin{aligned} \mathcal{L}(\boldsymbol{\Theta} \mid \boldsymbol{Y}_{0:M}) &\propto p(\boldsymbol{Y}_{0:M} \mid \boldsymbol{Z}_{1:N} = \boldsymbol{0}, \boldsymbol{\Theta}) \\ &= \int p(\boldsymbol{Y}_{0:M} \mid \boldsymbol{X}_{0:N}, \boldsymbol{\phi}) p(\boldsymbol{X}_{1:N} \mid \boldsymbol{Z}_{1:N} = \boldsymbol{0}, \boldsymbol{\theta}, \boldsymbol{\eta}) \, \mathrm{d}\boldsymbol{X}_{1:N}. \end{aligned} \tag{2.30}$$

We present several parameter inference methods within the Bayesian filtering framework of Tronarp et al. (2018).

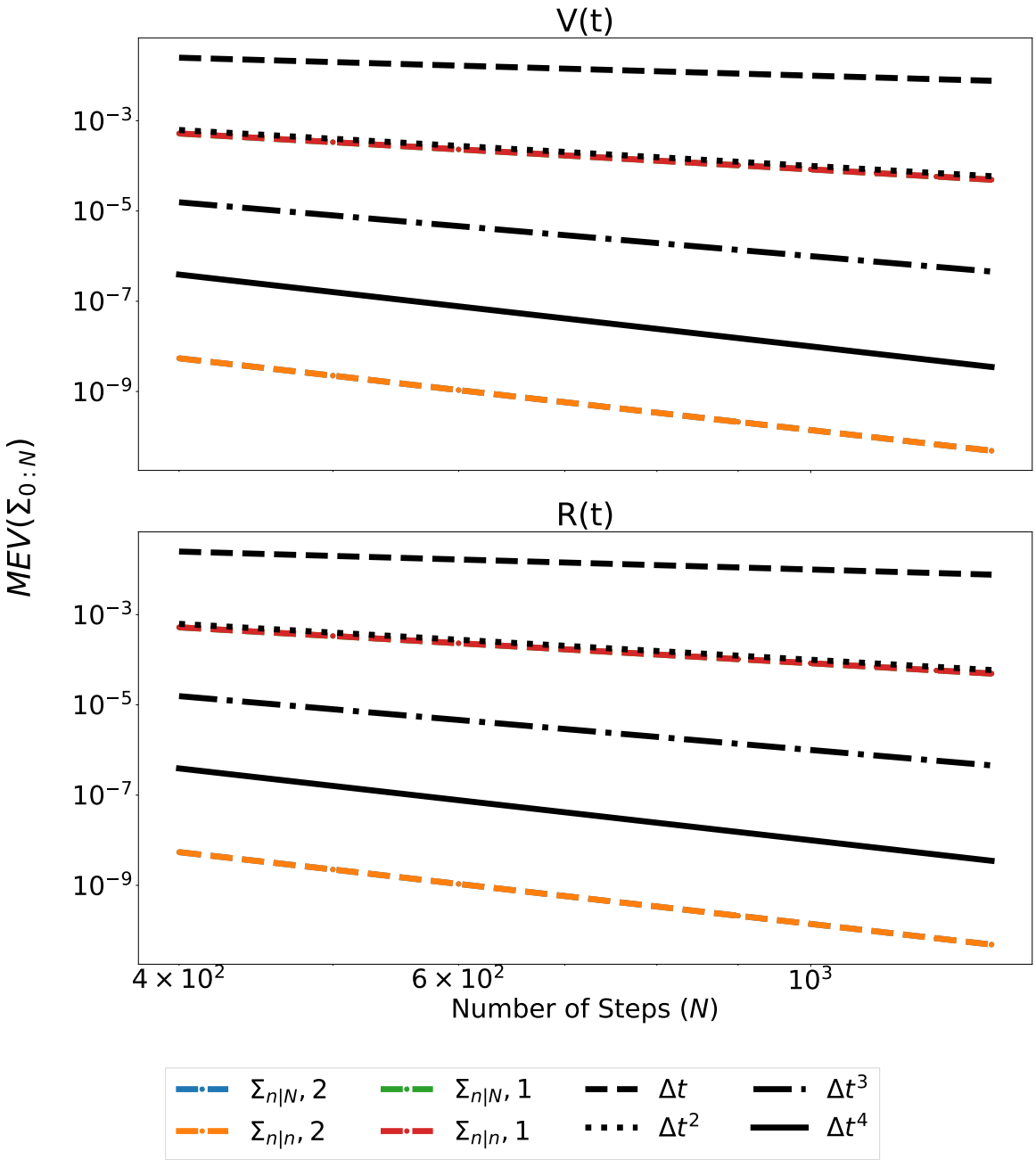

Figure 2.3: Convergence rates for the Kalman updated variance and the RTS smoothed variance using independent $\{1, 2\}$-times IBM prior.

2.5.1 Basic Method

A basic approximation to the likelihood function (2.30) is to use the RTS smoother mean from the surrogate model (2.29), $\boldsymbol{\mu}_{0:N|N}(\boldsymbol{\theta}, \boldsymbol{\eta}) = E_{\text{lin}}[\boldsymbol{X}_{0:N} \mid \boldsymbol{Z}_{1:N}, \boldsymbol{\theta}, \boldsymbol{\eta}]$, and directly substitute it into the observation model (2.28),

$$\hat{\mathcal{L}}(\boldsymbol{\Theta} \mid \boldsymbol{Y}_{0:M}) = \prod_{i=0}^{M} p(\boldsymbol{Y}_i \mid \boldsymbol{X}_{n(i)} = \boldsymbol{\mu}_{n(i)|N}(\boldsymbol{\theta}, \boldsymbol{\eta}), \boldsymbol{\phi}). \tag{2.31}$$

A similar approach proposed in Schober et al. (2019) is to use the Kalman filter mean without the backward smoothing pass. However, neither approximation propagates the uncertainty in the probabilistic ODE solver to the calculation of the likelihood.

2.5.2 Physics-Enhanced Regression in Initial Value problems

Fenrir (Tronarp et al., 2022) is a recently proposed method which extends a likelihood estimation introduced in Kersting et al. (2020a) to compute the likelihood function of (2.30). Fenrir assumes that the observation model of (2.28) is Gaussian of the form

$$\boldsymbol{Y}_i \overset{\text{ind}}{\sim} \text{Normal}(\boldsymbol{D}_i^{(\boldsymbol{\phi})} \boldsymbol{X}_{n(i)}, \boldsymbol{\Omega}_i^{(\boldsymbol{\phi})}). \tag{2.32}$$

where $\boldsymbol{D}_i^{(\boldsymbol{\phi})}$ and $\boldsymbol{\Omega}_i^{(\boldsymbol{\phi})}$ are coefficient and variance matrices, respectively, which may depend on the noise parameters $\boldsymbol{\phi}$.

Fenrir consists of three major steps. The first step begins with the surrogate model in (2.29) and applies the Kalman filter to perform a full forward pass. At this stage, we have the distribution $p_{\text{lin}}(\boldsymbol{X}_n \mid \boldsymbol{Z}_{1:n}, \boldsymbol{\theta}, \boldsymbol{\eta})$ for $n = 0, \ldots, N$. Rather than completing a backward pass to compute the smoothing distribution $p_{\text{lin}}(\boldsymbol{X}_{1:N} \mid \boldsymbol{Z}_{1:N}, \boldsymbol{\theta}, \boldsymbol{\eta})$, Tronarp et al. (2022) suggest leveraging the fact that a Gaussian Markov process moving forward in time also remains a Gaussian Markov process when reversed. Consequently, the smoothing distribution $p_{\text{lin}}(\boldsymbol{X}_{1:N} \mid \boldsymbol{Z}_{1:N}, \boldsymbol{\theta}, \boldsymbol{\eta})$ can be simulated using

$$\begin{aligned} \boldsymbol{X}_N &\sim \text{Normal}(\boldsymbol{b}_N, \boldsymbol{C}_N) \\ \boldsymbol{X}_n \mid \boldsymbol{X}_{n+1} &\sim \text{Normal}(\boldsymbol{A}_n \boldsymbol{X}_{n+1} + \boldsymbol{b}_n, \boldsymbol{C}_n), \end{aligned} \tag{2.33}$$

where the quantities $\boldsymbol{A}_{0:N-1}$, $\boldsymbol{b}_{0:N}$, and $\boldsymbol{C}_{0:N}$ are parameters to a non-homogeneous Markov process.

The second step of the Fenrir algorithm employs the RTS sampler (Rauch et al., 1965) to recursively compute the parameters of the model (2.33), given by

$$\begin{aligned} \boldsymbol{A}_n &= \boldsymbol{\Sigma}_{n|n}\boldsymbol{Q}_{\boldsymbol{\eta}}^\top\boldsymbol{\Sigma}_{n+1|n}^{-1} \\ \boldsymbol{b}_n &= \boldsymbol{\mu}_{n|n} - \boldsymbol{A}_n\boldsymbol{\mu}_{n+1|n} \\ \boldsymbol{C}_n &= \boldsymbol{\Sigma}_{n|n} - \boldsymbol{A}_n\boldsymbol{Q}_{\boldsymbol{\eta}}\boldsymbol{\Sigma}_{n|n} \end{aligned} \tag{2.34}$$

where at the final time point t_N, the values are initialized as $\boldsymbol{b}_N = \boldsymbol{\mu}_{N|N}$ and $\boldsymbol{C}_n = \boldsymbol{\Sigma}_{N|N}$ corresponding to the Kalman filter mean and variance.

The final step involves augmenting the observation model from (2.32) into the backward Gaussian Markov process, leading to the model

$$\begin{aligned} \boldsymbol{X}_N &\sim \text{Normal}(\boldsymbol{b}_N, \boldsymbol{C}_N) \\ \boldsymbol{X}_n \mid \boldsymbol{X}_{n+1} &\sim \text{Normal}(\boldsymbol{A}_n\boldsymbol{X}_{n+1} + \boldsymbol{b}_n, \boldsymbol{C}_n) \\ \boldsymbol{Y}_i &\overset{\text{ind}}{\sim} \text{Normal}(\boldsymbol{D}_i^{(\boldsymbol{\phi})}\boldsymbol{X}_{n(i)}, \boldsymbol{\Omega}_i^{(\boldsymbol{\phi})}). \end{aligned} \tag{2.35}$$

It is important to note that the ODE solution time grid may differ from the observation time grid. In such cases, at time points where no observation is available, the update phase is skipped, and the updated mean and variance are set equal to their predicted values. Intuitively, if no new information is observed, the predicted estimates remain the optimal estimates.

Fenrir approximates the likelihood function in (2.30) using the prediction error decomposition (Schweppe, 1965), expressed as

$$p(\boldsymbol{Y}_{0:M} \mid \boldsymbol{Z}_{1:N} = \boldsymbol{0}, \boldsymbol{\Theta}) \approx \prod_{i=0}^{M} \text{Normal}(\boldsymbol{Y}_i; \boldsymbol{\mu}_i^{\boldsymbol{Y}}, \boldsymbol{\Sigma}_i^{\boldsymbol{Y}}). \tag{2.36}$$

where $\boldsymbol{\mu}_i^{\boldsymbol{Y}}$ and $\boldsymbol{\Sigma}_i^{\boldsymbol{Y}}$ can be computed alongside the Kalman filter by

$$\begin{aligned} \boldsymbol{\mu}_i^{\boldsymbol{Y}} &= \boldsymbol{D}_i^{(\boldsymbol{\phi})}\boldsymbol{\mu}_{n(i)|n(i)+1} \\ \boldsymbol{\Sigma}_i^{\boldsymbol{Y}} &= \boldsymbol{D}_i^{(\boldsymbol{\phi})}\boldsymbol{\Sigma}_{n(i)|n(i)+1}\boldsymbol{D}_i^{(\boldsymbol{\phi})\top} + \boldsymbol{\Omega}_i^{(\boldsymbol{\phi})}. \end{aligned} \tag{2.37}$$

In this context, $\boldsymbol{\mu}_{n(i)|n(i)+1}$ and $\boldsymbol{\Sigma}_{n(i)|n(i)+1}$ represent the predicted mean and variance, as the Kalman filter operates backward in time from t_N to t_0 within the model described in (2.35). Algorithm 2.2 outlines the key steps of the Fenrir method for approximating the likelihood function in (2.30), where the function `normal_logpdf`() on Line 8 is the log-pdf

Algorithm 2.2 The Fenrir probabilistic ODE likelihood approximation.

1: **procedure** `fenrir`$(\boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{X}, t), \boldsymbol{v}_{\boldsymbol{\theta}}, \boldsymbol{Q}_{\boldsymbol{\eta}}, \boldsymbol{R}_{\boldsymbol{\eta}}, \boldsymbol{Y}_{0:M}, \boldsymbol{D}^{(\phi)}_{0:M}, \boldsymbol{\Omega}^{(\phi)}_{0:M})$
2: $\quad \ell \leftarrow 0$ ▷ Initialization
3: $\quad i \leftarrow M - 1$ ▷ Used to map $t_{n(i)}$ to t'_i
4: $\quad \boldsymbol{\mu}_{0|0}, \ldots, \boldsymbol{\mu}_{N|N}, \boldsymbol{\Sigma}_{0|0}, \ldots, \boldsymbol{\Sigma}_{N|N} \leftarrow$ `kalman_ode_forward`$(\boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{X}, t), \boldsymbol{v}_{\boldsymbol{\theta}}, \boldsymbol{Q}_{\boldsymbol{\eta}}, \boldsymbol{R}_{\boldsymbol{\eta}})$
5: ▷ Lines 6-20 compute $\log p(\boldsymbol{Y}_{0:M} \mid \boldsymbol{Z}_{1:N} = \boldsymbol{0}, \boldsymbol{\Theta})$
6: $\quad \boldsymbol{A}_N, \boldsymbol{b}_N, \boldsymbol{C}_N \leftarrow \boldsymbol{0}, \boldsymbol{\mu}_{N|N}, \boldsymbol{\Sigma}_{N|N}$
7: $\quad \boldsymbol{\mu}_N, \boldsymbol{\Sigma}_N \leftarrow$ `kalman_forecast`$(\boldsymbol{\mu}_{N|N}, \boldsymbol{\Sigma}_{N|N}, \boldsymbol{0}, \boldsymbol{D}_M, \boldsymbol{\Omega}_M)$
8: $\quad \ell \leftarrow \ell +$ `normal_logpdf`$(\boldsymbol{y}_N; \boldsymbol{\mu}_N, \boldsymbol{\Sigma}_N)$
9: $\quad \boldsymbol{\mu}'_{N|N}, \boldsymbol{\Sigma}'_{N|N} \leftarrow$ `kalman_update`$(\boldsymbol{\mu}_{N|N}, \boldsymbol{\Sigma}_{N|N}, \boldsymbol{Y}_M, \boldsymbol{0}, \boldsymbol{D}_M, \boldsymbol{\Omega}_M)$
10: $\quad$ **for** $n = N - 1 : 0$ **do**
11: $\quad\quad \boldsymbol{A}_n, \boldsymbol{b}_n, \boldsymbol{C}_n \leftarrow$ `kalman_cond`$(\boldsymbol{\mu}_{n|n}, \boldsymbol{\Sigma}_{n|n}, \boldsymbol{\mu}_{n+1|n}, \boldsymbol{\Sigma}_{n+1|n}, \boldsymbol{Q})$
12: $\quad\quad \boldsymbol{\mu}'_{n|n+1}, \boldsymbol{\Sigma}'_{n|n+1} \leftarrow$ `kalman_predict`$(\boldsymbol{\mu}'_{n+1|n+1}, \boldsymbol{\Sigma}'_{n+1|n+1}, \boldsymbol{b}_n, \boldsymbol{A}_n, \boldsymbol{C})$
13: $\quad\quad$ **if** $t_n = t_{n(i)}$ **then**
14: $\quad\quad\quad \boldsymbol{\mu}_n, \boldsymbol{\Sigma}_n \leftarrow$ `kalman_forecast`$(\boldsymbol{\mu}'_{n|n+1}, \boldsymbol{\Sigma}'_{n|n+1}, \boldsymbol{0}, \boldsymbol{D}_i, \boldsymbol{\Omega}_i)$
15: $\quad\quad\quad \ell \leftarrow \ell +$ `normal_logpdf`$(\boldsymbol{Y}_i; \boldsymbol{\mu}_n, \boldsymbol{\Sigma}_n)$
16: $\quad\quad\quad \boldsymbol{\mu}'_{n|n}, \boldsymbol{\Sigma}'_{n|n} \leftarrow$ `kalman_update`$(\boldsymbol{\mu}'_{n|n+1}, \boldsymbol{\Sigma}'_{n|n+1}, \boldsymbol{y}_n, \boldsymbol{0}, \boldsymbol{D}_i, \boldsymbol{\Omega}_i)$
17: $\quad\quad\quad i \leftarrow i - 1$
18: $\quad\quad$ **else**
19: $\quad\quad\quad \boldsymbol{\mu}'_{n|n}, \boldsymbol{\Sigma}'_{n|n} \leftarrow \boldsymbol{\mu}'_{n|n+1}, \boldsymbol{\Sigma}'_{n|n+1}$
20: $\quad$ **return** ℓ

of a multivariate normal. The related Kalman functions are also available in pseudo-code in Appendix A.2 for reference.

Additionally, if the ODE solution posterior, $p_{\text{lin}}(\boldsymbol{X}_{1:N} \mid \boldsymbol{Y}_{0:M}, \boldsymbol{Z}_{1:N}, \boldsymbol{\Theta})$, is desired instead of the likelihood, then both the Kalman filter and RTS smoother must be applied to (2.35). In this case, the RTS smoother operates in a forward in time fashion because the Kalman filter is already reversed in the model (2.35). Since the Kalman algorithm scales linearly with both the ODE dimensions and discretization points, Fenrir does so as well.

2.5.3 Marginal MCMC

In principle, the basic approximation can utilize any of the model interrogation methods listed in Table 2.1. However, the interrogation method of Chkrebtii et al. (2016) generates $\boldsymbol{a}_{n+1}$ stochastically and provides a nonzero variance estimate $\boldsymbol{V}_{n+1}$ for this quantity. As a

result, applying the basic approximation would not account for the additional uncertainty introduced at this step. To address this, Chkrebtii et al. (2016) adopt a Bayesian approach, where, given a parameter prior $\pi(\boldsymbol{\Theta})$ and arbitrary observation model (2.28), they present a marginal MCMC algorithm to estimate

$$\begin{aligned}\hat{p}(\boldsymbol{\Theta}, \boldsymbol{X}_{1:N} \mid \boldsymbol{Y}_{0:M}, \boldsymbol{Z}_{1:N} = \boldsymbol{0}) \propto \pi(\boldsymbol{\Theta}) \times p(\boldsymbol{Y}_{0:M} \mid \boldsymbol{X}_{0:N}, \boldsymbol{\phi}) \\ \times \int \hat{p}_{\text{lin}}(\boldsymbol{X}_{1:N}, \boldsymbol{\mathcal{I}}_{1:N} \mid \boldsymbol{Z}_{1:N} = \boldsymbol{0}, \boldsymbol{\theta}, \boldsymbol{\eta}) \, \mathrm{d}\boldsymbol{\mathcal{I}}_{1:N},\end{aligned} \tag{2.38}$$

where $\hat{p}_{\text{lin}}(\boldsymbol{X}_{1:N}, \boldsymbol{\mathcal{I}}_{1:N} \mid \boldsymbol{Z}_{1:N} = \boldsymbol{0}, \boldsymbol{\theta}, \boldsymbol{\eta})$ corresponds to the parameter-dependent surrogate model (2.29) with stochastic model interrogation terms $\boldsymbol{\mathcal{I}}_{1:N} = (\boldsymbol{a}_{1:N}, \boldsymbol{B}_{1:N}, \boldsymbol{V}_{1:N})$.

The specific MCMC algorithm used is the Metropolis-Hastings algorithm, with the update rule detailed in Algorithm 2.3. In particular, the function `kalman_ode_sim` is called on Line 3 to generate a sample solution, as described in Algorithm 2.1. The term "marginal" refers to the fact that $\hat{p}_{\text{lin}}(\boldsymbol{X}_{1:N}, \boldsymbol{\mathcal{I}}_{1:N} \mid \boldsymbol{Z}_{1:N} = \boldsymbol{0}, \boldsymbol{\theta}, \boldsymbol{\eta})$ appears both in the target posterior density and the Markov transition kernel, causing it to cancel out entirely in the Metropolis-Hastings acceptance rate.

Algorithm 2.3 Marginal MCMC algorithm of Chkrebtii et al. (2016).

1: **procedure** `marginal_mcmc_step`$(\boldsymbol{\Theta}^{(\text{curr})}, \boldsymbol{X}_{0:N}^{(\text{curr})}, \boldsymbol{Y}_{0:M})$
2: $\quad \boldsymbol{\Theta}^{(\text{prop})} \sim q(\boldsymbol{\Theta} \mid \boldsymbol{\Theta}^{(\text{curr})})$
3: $\quad \boldsymbol{X}_{0:N}^{(\text{prop})} \leftarrow$ `kalman_ode_sim`$(f_{\boldsymbol{\theta}^{(\text{prop})}}(\boldsymbol{X}, t), \boldsymbol{v}_{\boldsymbol{\theta}^{(\text{prop})}}, \boldsymbol{Q}_{\boldsymbol{\eta}^{(\text{prop})}}, \boldsymbol{R}_{\boldsymbol{\eta}^{(\text{prop})}})$
4: $\quad \rho = \dfrac{p(\boldsymbol{Y}_{0:M} \mid \boldsymbol{X}_{n(0:M)}^{(\text{prop})}, \boldsymbol{\eta}^{(\text{prop})}) \times \pi(\boldsymbol{\Theta}^{(\text{prop})}) / q(\boldsymbol{\Theta}^{(\text{prop})} \mid \boldsymbol{\Theta}^{(\text{curr})})}{p(\boldsymbol{Y}_{0:M} \mid \boldsymbol{X}_{n(0:M)}^{(\text{curr})}, \boldsymbol{\eta}^{(\text{curr})}) \times \pi(\boldsymbol{\Theta}^{(\text{curr})}) / q(\boldsymbol{\Theta}^{(\text{curr})} \mid \boldsymbol{\Theta}^{(\text{prop})})}$
5: $\quad v \sim \text{Uniform}(0, 1)$
6: $\quad$ **if** $v < \min(\rho, 1)$ **then**
7: $\quad\quad \boldsymbol{\Theta}^{(\text{curr})}, \boldsymbol{X}_{0:N}^{(\text{curr})}, \rho^{(\text{curr})} \leftarrow \boldsymbol{\Theta}^{(\text{prop})}, \boldsymbol{X}_{0:N}^{(\text{prop})}, \rho^{(\text{prop})}$
8: $\quad$ **return** $\boldsymbol{\Theta}^{(\text{curr})}, \boldsymbol{X}_{0:N}^{(\text{curr})}, \rho^{(\text{curr})}$

2.5.4 MAGI

Our adapted MAGI is a method for performing Bayesian inference with the exact posterior

$$p(\boldsymbol{\Theta} \mid \boldsymbol{Z}_{1:N} = \boldsymbol{0}, \boldsymbol{Y}_{0:M}), \tag{2.39}$$

corresponding to the nonlinear SSM

$$\begin{aligned} \boldsymbol{X}_{n+1} \mid \boldsymbol{X}_n &\sim \text{Normal}(\boldsymbol{Q}_{\boldsymbol{\eta}} \boldsymbol{X}_n, \boldsymbol{R}_{\boldsymbol{\eta}}) \\ \boldsymbol{Z}_{n+1} &\overset{\text{ind}}{\sim} \text{Normal}(\dot{\boldsymbol{x}}_{n+1} - \boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{x}_{n+1}, t_{n+1}), \boldsymbol{V}_{n+1}), \end{aligned} \tag{2.40}$$

with $\boldsymbol{V}_{n+1} = \boldsymbol{0}$. Unlike previous approaches that approximate the likelihood using the surrogate SSM (2.29), the adapted MAGI directly uses the original SSM, thereby preserving the full nonlinearity of the system. This approach also differs from the original MAGI algorithm proposed by Yang et al. (2021), which assumes a Gaussian process prior on the solution process $\boldsymbol{X}_{0:N}$, but not necessarily a Markov one. In contrast, our approach additionally assumes that the prior on $\boldsymbol{X}_{0:N}$ has the Markov property, which substantially reduces the computational burden. We demonstrate how the adapted MAGI posterior can be efficiently and scalably computed within the Bayesian filtering paradigm of Tronarp et al. (2018).

The MAGI approach provides a closed form for the joint posterior

$$\begin{aligned} p(\boldsymbol{\Theta}, \boldsymbol{X}_{1:N} \mid \boldsymbol{Z}_{1:N} = \boldsymbol{0}, \boldsymbol{Y}_{0:M}) &\propto p(\boldsymbol{\Theta}, \boldsymbol{X}_{1:N}, \boldsymbol{Z}_{1:N} = \boldsymbol{0}, \boldsymbol{Y}_{0:M}) \\ &= p(\boldsymbol{\Theta}) \times p(\boldsymbol{X}_{1:N}, \boldsymbol{Z}_{1:N} = \boldsymbol{0} \mid \boldsymbol{\Theta}) \times p(\boldsymbol{Y}_{0:M} \mid \boldsymbol{X}_{0:N}, \boldsymbol{\phi}). \end{aligned} \tag{2.41}$$

The first and third terms on the right-hand side of (2.41) are straightforward to compute. However, note that $p(\boldsymbol{X}_{1:N}, \boldsymbol{Z}_{1:N} \mid \boldsymbol{\Theta})$ is a degenerate distribution, since $\boldsymbol{Z}_{1:N} = \dot{\boldsymbol{x}}_{1:N} - \boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{x}_{1:N}, t_{1:N})$ is uniquely determined by $\boldsymbol{X}_{1:N}$. A major contribution of Yang et al. (2021) is the derivation of the density for this degenerate distribution. We present this derivation in a slightly more general setting below.

In order to evaluate the second term, assume that $\boldsymbol{X}_{0:N}$ can be decomposed as $\boldsymbol{X}_{0:N} = (\boldsymbol{U}_{0:N}, \boldsymbol{W}_{0:N})$, where $\boldsymbol{W}_{0:N}$ is a proper subset of $\boldsymbol{X}_{0:N}$ and $\boldsymbol{U}_{0:N}$ represents the remaining components of $\boldsymbol{X}_{0:N}$, such that $\boldsymbol{W}_n = \boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{U}_n, t_n)$. For example, consider a simple univariate first-order ODE

$$x^{(1)}(t) = \boldsymbol{f}_{\boldsymbol{\theta}}(x^{(0)}(t), t), \tag{2.42}$$

along with the observation model

$$\boldsymbol{Y}_i \overset{\text{ind}}{\sim} p(\boldsymbol{Y}_i \mid x^{(0)}(t_i'), \boldsymbol{\phi}). \tag{2.43}$$

We assume a 2-times IBM prior on the solution process such that, at time step n, we have

$$\boldsymbol{X}_n = \begin{bmatrix} x_n^{(0)} \\ x_n^{(1)} \\ x_n^{(2)} \end{bmatrix}, \tag{2.44}$$

with the corresponding decomposition

$$\boldsymbol{W}_n = \begin{bmatrix} x_n^{(1)} \end{bmatrix}, \qquad \boldsymbol{U}_n = \begin{bmatrix} x_n^{(0)} \\ x_n^{(2)} \end{bmatrix}. \tag{2.45}$$

Rather than directly deriving the density of the degenerate distribution $p(\boldsymbol{X}_{1:N}, \boldsymbol{Z}_{1:N} \mid \boldsymbol{\Theta})$, notice that $(\boldsymbol{X}_{0:N}, \boldsymbol{Z}_{1:N})$ can be fully characterized by $(\boldsymbol{U}_{0:N}, \boldsymbol{Z}_{1:N})$. Therefore, we instead consider the non-degenerate joint distribution

$$p(\boldsymbol{U}_{1:N}, \boldsymbol{Z}_{1:N} \mid \boldsymbol{\Theta}), \tag{2.46}$$

which can be obtained via a change of variables from $\boldsymbol{X}_{0:N} = (\boldsymbol{U}_{0:N}, \boldsymbol{W}_{0:N})$. To carry out this transformation, we compute the determinant of the Jacobian for the mapping $(\boldsymbol{U}_{0:N}, \boldsymbol{W}_{0:N}) \to (\boldsymbol{U}_{0:N}, \boldsymbol{Z}_{1:N})$, given by

$$\begin{vmatrix} \dfrac{\mathrm{d}\boldsymbol{U}_{0:N}}{\mathrm{d}\boldsymbol{U}_{0:N}} & \dfrac{\mathrm{d}\boldsymbol{Z}_{1:N}}{\mathrm{d}\boldsymbol{U}_{0:N}} \\ \dfrac{\mathrm{d}\boldsymbol{U}_{0:N}}{\mathrm{d}\boldsymbol{W}_{0:N}} & \dfrac{\mathrm{d}\boldsymbol{Z}_{1:N}}{\mathrm{d}\boldsymbol{W}_{0:N}} \end{vmatrix} = \begin{vmatrix} \boldsymbol{I} & \dfrac{\mathrm{d}\boldsymbol{Z}_{1:N}}{\mathrm{d}\boldsymbol{U}_{0:N}} \\ \boldsymbol{0} & \boldsymbol{I} \end{vmatrix} = 1. \tag{2.47}$$

Thus, we obtain the relationship

$$\begin{aligned} p(\boldsymbol{U}_{1:N}, \boldsymbol{Z}_{1:N} \mid \boldsymbol{\Theta}) &= p(\boldsymbol{X}_{1:N} \mid \boldsymbol{\Theta}) \\ &= \prod_{n=1}^{N} \varphi(\boldsymbol{X}(\boldsymbol{U}_n, \boldsymbol{Z}_n) - \boldsymbol{Q}_{\boldsymbol{\eta}} \boldsymbol{X}(\boldsymbol{U}_{n-1}, \boldsymbol{Z}_{n-1}) \mid \boldsymbol{R}_{\boldsymbol{\eta}}), \end{aligned} \tag{2.48}$$

where $\boldsymbol{X}(\boldsymbol{U}_n, \boldsymbol{Z}_n) = (\boldsymbol{U}_n, \boldsymbol{Z}_n + \boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{U}_n, t_n))$ and $\varphi(\boldsymbol{z} \mid \boldsymbol{\Sigma})$ denotes the density of a multivariate normal with mean zero and variance $\boldsymbol{\Sigma}$. In other words, $p(\boldsymbol{U}_{1:N}, \boldsymbol{Z}_{1:N} \mid \boldsymbol{\Theta})$ is the density of the Gaussian Markov prior in (2.40), evaluated at $\boldsymbol{X}_{0:N} = \boldsymbol{X}(\boldsymbol{U}_{0:N}, \boldsymbol{Z}_{1:N})$. Rewriting the joint posterior from (2.41) in terms of the transformed variables $(\boldsymbol{U}_{0:N}, \boldsymbol{Z}_{1:N})$, we obtain

$$\begin{aligned} p(\boldsymbol{\Theta}, \boldsymbol{U}_{1:N} \mid \boldsymbol{Z}_{1:N} = \boldsymbol{0}, \boldsymbol{Y}_{0:M}) &\propto p(\boldsymbol{\Theta}, \boldsymbol{U}_{1:N}, \boldsymbol{Z}_{1:N} = \boldsymbol{0}, \boldsymbol{Y}_{0:M}) \\ &= \pi(\boldsymbol{\Theta}) \times p(\boldsymbol{U}_{1:N}, \boldsymbol{Z}_{1:N} = \boldsymbol{0} \mid \boldsymbol{\Theta}) \times p(\boldsymbol{Y}_{0:M} \mid \boldsymbol{X}_{0:N}, \boldsymbol{\phi}), \end{aligned} \tag{2.49}$$

where the second term is given by (2.48).

Sampling from the MAGI posterior in (2.49) is done via MCMC, from which we obtain samples from $p(\boldsymbol{\Theta} \mid \boldsymbol{Z}_{1:N} = \boldsymbol{0}, \boldsymbol{Y}_{0:M})$ as a byproduct. However, it is often possible to

significantly reduce the number of variables in the MCMC sampler. That is, consider the smallest subset $\tilde{\boldsymbol{U}}_{0:N} \subseteq \boldsymbol{U}_{0:N}$ such that,

$$\begin{aligned} \boldsymbol{f}_{\boldsymbol{\theta}}(\tilde{\boldsymbol{U}}_{0:N}, t_{0:N}) &= \boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{U}_{0:N}, t_{0:N}), \\ p(\boldsymbol{Y}_{0:M} \mid \tilde{\boldsymbol{X}}_{0:N}, \boldsymbol{\phi}) &= p(\boldsymbol{Y}_{0:M} \mid \boldsymbol{X}_{0:N}, \boldsymbol{\phi}), \end{aligned} \tag{2.50}$$

where $\tilde{\boldsymbol{X}}_{0:N} = (\tilde{\boldsymbol{U}}_{0:N}, \boldsymbol{W}_{0:N})$. In the example with the ODE defined in (2.42) and observation model in (2.43), the smallest subset at each time step n is

$$\tilde{\boldsymbol{U}}_n = \begin{bmatrix} x_n^{(0)} \end{bmatrix}, \qquad \tilde{\boldsymbol{X}}_n = \begin{bmatrix} x_n^{(0)} \\ x_n^{(1)} \end{bmatrix}. \tag{2.51}$$

Using the smallest subset, the modified MAGI posterior distribution can be computed as

$$\begin{aligned} p(\boldsymbol{\Theta}, \tilde{\boldsymbol{U}}_{1:N} \mid \boldsymbol{Y}_{0:M}, \boldsymbol{Z}_{1:N} = \boldsymbol{0}) &\propto \pi(\boldsymbol{\Theta}) \times p(\tilde{\boldsymbol{X}}_{1:N} \mid \boldsymbol{\Theta}) \times p(\boldsymbol{Y}_{0:M} \mid \tilde{\boldsymbol{X}}_{0:N}, \boldsymbol{\phi}), \\ &= \pi(\boldsymbol{\Theta}) \times p(\tilde{\boldsymbol{X}}(\tilde{\boldsymbol{U}}_{1:N}) \mid \boldsymbol{\eta}) \times p(\boldsymbol{Y}_{0:M} \mid \tilde{\boldsymbol{X}}(\tilde{\boldsymbol{U}}_{0:N}), \boldsymbol{\phi}), \end{aligned} \tag{2.52}$$

where $\tilde{\boldsymbol{X}}(\tilde{\boldsymbol{U}}_n) = (\tilde{\boldsymbol{U}}_n, \boldsymbol{f}_{\boldsymbol{\theta}}(\tilde{\boldsymbol{U}}_n, t_n))$. That is, the second term in (2.52) is the density induced by the Gaussian Markov process prior from (2.40) on $\tilde{\boldsymbol{X}}_{0:N} = (\tilde{\boldsymbol{U}}_{0:N}, \boldsymbol{W}_{0:N})$, evaluated at $\tilde{\boldsymbol{X}}(\tilde{\boldsymbol{U}}_{0:N})$.

The sampled $\tilde{\boldsymbol{X}}_{1:N}$ in (2.52) are typically not Markov. However, since $\tilde{\boldsymbol{X}}_{1:N}$ is a subset of the Gaussian Markov random variables $\boldsymbol{X}_{1:N}$, we can use Kalman recursions to efficiently compute the joint density $p(\tilde{\boldsymbol{X}}_{1:N} \mid \boldsymbol{\eta})$ with linear time complexity $\mathcal{O}(N)$. Thus, the MAGI posteriors (2.49) and (2.52) for the full and subset variables $\boldsymbol{U}_{1:N}$ and $\tilde{\boldsymbol{U}}_{1:N}$ have the same computational complexity per evaluation, but the latter contains $c \times N$ fewer random variables, where c is a function of the number of system variables d and the number of additional derivatives $q - p$ in the IBM prior (2.5). This can be a significant advantage for speeding up the convergence of the MCMC sampler on (2.52).

In practice, the length of the solution process N is often much greater than the number of observations M, in which case the modified MAGI posterior in (2.52) is dominated by the second term (Yang et al., 2021). This occurs because the density of $p(\tilde{\boldsymbol{X}}_{1:N} \mid \boldsymbol{\eta})$ includes many more terms than the likelihood $p(\boldsymbol{Y}_{0:M} \mid \tilde{\boldsymbol{X}}_{0:N}, \boldsymbol{\phi})$. To address this imbalance, Yang et al. (2021) suggest tempering the density $p(\tilde{\boldsymbol{X}}_{1:N} \mid \boldsymbol{\eta})^{1/\beta}$ by a hyperparameter β, referred to as the prior temperature, to better balance the contributions of the density and the likelihood. The β-modulated MAGI posterior is thus

$$\begin{aligned} p_\beta(\boldsymbol{\Theta}, \tilde{\boldsymbol{U}}_{1:N} \mid \boldsymbol{Z}_{1:N}, \boldsymbol{Y}_{0:M}) &\propto p_\beta(\boldsymbol{\Theta}, \tilde{\boldsymbol{U}}_{1:N}, \boldsymbol{Z}_{1:N} = \boldsymbol{0}, \boldsymbol{Y}_{0:M}) \\ &= \pi(\boldsymbol{\Theta}) \times p(\tilde{\boldsymbol{X}}(\tilde{\boldsymbol{U}}_{1:N}) \mid \boldsymbol{\eta})^{1/\beta} \times p(\boldsymbol{Y}_{0:M} \mid \tilde{\boldsymbol{X}}(\tilde{\boldsymbol{U}}_{0:N}), \boldsymbol{\phi}). \end{aligned} \tag{2.53}$$

The exact steps for computing (2.53) are presented in Algorithm 2.4, where the function `normal_logpdf`() on Line 9 is the log probability density function of a multivariate normal. The related Kalman functions can be found in Algorithm A.1 in Appendix A.2.

Algorithm 2.4 MAGI algorithm of (Yang et al., 2021) for computing the β-modulated posterior (2.53) with Gaussian Markov process prior.

1: **procedure** `magi_logdens`$(\tilde{\boldsymbol{U}}_{0:N}, \boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{X}, t), \boldsymbol{v}_{\boldsymbol{\theta}}, \boldsymbol{Q}_{\boldsymbol{\eta}}, \boldsymbol{R}_{\boldsymbol{\eta}}, \boldsymbol{Y}_{0:M}, \beta)$
2: $\quad \boldsymbol{\mu}_{0|0}, \Sigma_{0|0} \leftarrow \boldsymbol{v}_{\boldsymbol{\theta}}, \boldsymbol{0}$ ▷ Initialization
3: $\quad \ell \leftarrow 0$
4: $\quad \tilde{\boldsymbol{X}}_{0:N} \leftarrow (\tilde{\boldsymbol{U}}_{0:N}, \boldsymbol{f}_{\boldsymbol{\theta}}(\tilde{\boldsymbol{U}}_{0:N}, t_{0:N}))$
5: $\quad \tilde{\boldsymbol{W}} \leftarrow \begin{bmatrix} \boldsymbol{I}_{d\times q} & \boldsymbol{0}_{d\times p-q} \end{bmatrix}$ ▷ The dimensions are $\boldsymbol{X}_{d\times p}$ and $\tilde{\boldsymbol{X}}_{d\times q}$.
6: $\quad$ **for** $n = 1 : N$ **do**
7: $\quad\quad \boldsymbol{\mu}_{n|n-1}, \boldsymbol{\Sigma}_{n|n-1} \leftarrow$ `kalman_predict`$(\boldsymbol{\mu}_{n-1|n-1}, \boldsymbol{\Sigma}_{n-1|n-1}, \boldsymbol{0}, \boldsymbol{Q}_{\boldsymbol{\eta}}, \boldsymbol{R}_{\boldsymbol{\eta}})$
8: $\quad\quad \boldsymbol{\mu}_n, \boldsymbol{\Sigma}_n \leftarrow$ `kalman_forecast`$(\boldsymbol{\mu}_{n|n-1}, \boldsymbol{\Sigma}_{n|n-1}, \boldsymbol{0}, \tilde{\boldsymbol{W}}, \boldsymbol{0})$
9: $\quad\quad \ell \leftarrow \ell +$ `normal_logpdf`$(\tilde{\boldsymbol{X}}_n; \boldsymbol{\mu}_n, \boldsymbol{\Sigma}_n)$
10: $\quad\quad \boldsymbol{\mu}_{n|n}, \boldsymbol{\Sigma}_{n|n} \leftarrow$ `kalman_update`$(\boldsymbol{\mu}_{n|n-1}, \boldsymbol{\Sigma}_{n|n-1}, \tilde{\boldsymbol{X}}_n, \boldsymbol{0}, \tilde{\boldsymbol{W}}, \boldsymbol{0})$
11: ▷ Line 12 computes $\log p_\beta(\boldsymbol{\Theta}, \tilde{\boldsymbol{U}}_{1:N}, \boldsymbol{Z}_{1:N} = \boldsymbol{0}, \boldsymbol{Y}_{0:M})$
12: $\quad \ell \leftarrow \log \pi(\boldsymbol{\Theta}) + \frac{1}{\beta}\ell + \log p(\boldsymbol{Y}_{0:M} \mid \tilde{\boldsymbol{X}}_{0:N}, \boldsymbol{\phi})$
13: $\quad$ **return** ℓ

2.6 Numerical Experiments on Parameter Inference

We now return to the FN model from Section 2.4.1, where we previously focused on approximating the ODE. The model is governed by the system

$$\dot{V}(t) = c\Big(V(t) - \frac{V(t)^3}{3} + R(t)\Big), \quad \dot{R}(t) = -\frac{V(t) - a + bR(t)}{c}. \tag{2.54}$$

Our goal is to estimate the unknown parameters $\boldsymbol{\theta} = (a, b, c, V(0), R(0))$ where $a, b, c > 0$, using data $\boldsymbol{Y}_{0:M} = (\boldsymbol{Y}_0, \ldots, \boldsymbol{Y}_M)$ based on the observation model

$$\boldsymbol{Y}_i \overset{\text{ind}}{\sim} \text{Normal}(\boldsymbol{x}(t_i), \phi^2 \cdot \boldsymbol{I}_{2\times 2}). \tag{2.55}$$

We begin by simulating the data with $M = 40$, $t_i = i$, $\phi = 0.2$, and true parameters $\boldsymbol{\theta} = (0.2, 0.2, 3, -1, 1)$. To generate the ODE solution, we use a highly accurate deterministic

solver, namely, the Dormand-Prince 8/7 solver consisting of an 8th order Runge-Kutta method with 7th order step size adaptation (RKDP) (Dormand and Prince, 1980), as implemented in the **diffrax** library (Kidger, 2021). Observation noise is then added to the ODE solution using the default pseudo-random number generator in Python. The simulated ODE trajectory and noisy observations are displayed in Figure 2.4.

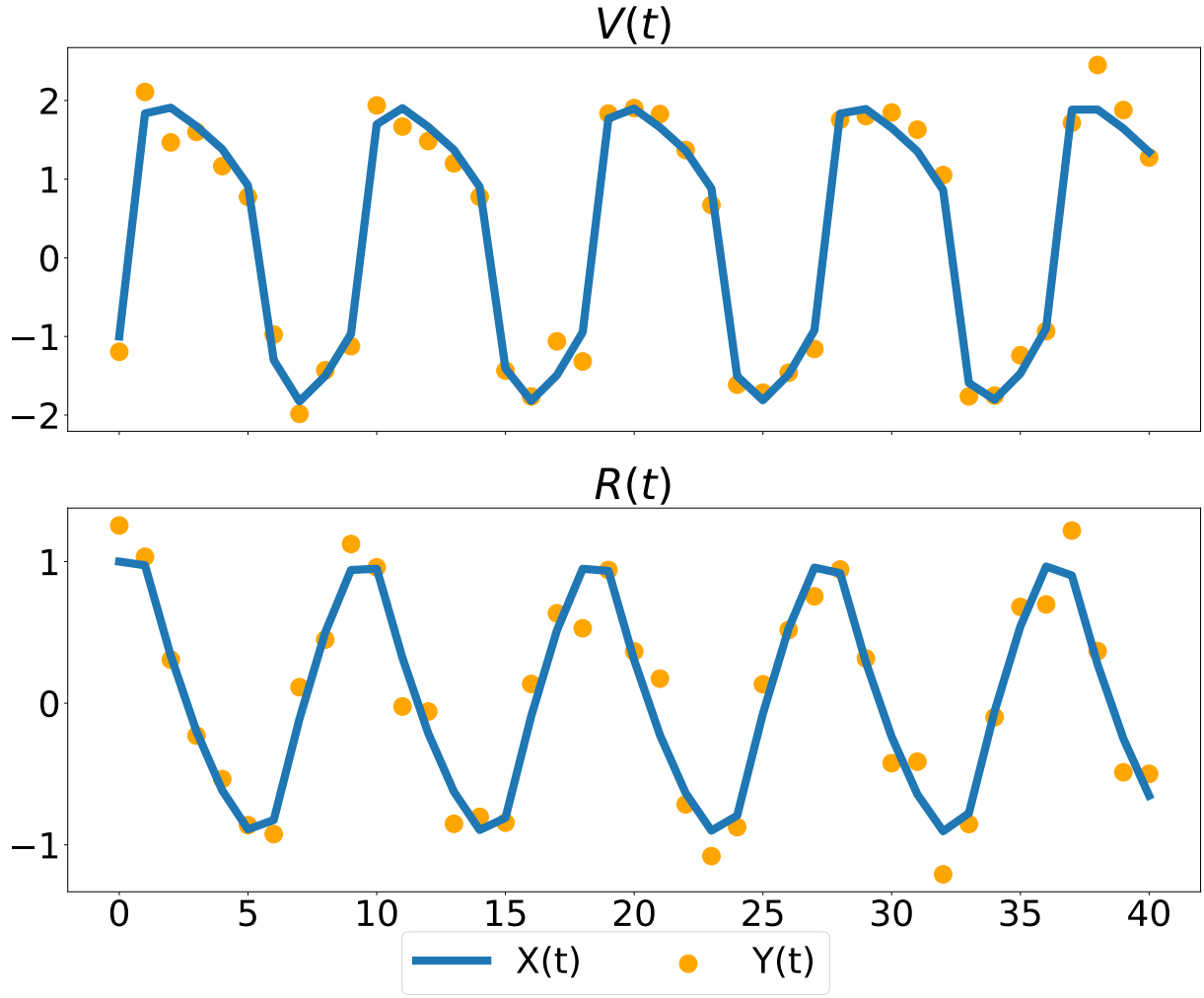

Figure 2.4: Simulated ODE and noisy observations of the FN model.

Having generated the data, we now compare the three parameter inference methods discussed in Section 2.5. Taking a Bayesian perspective, we assume a prior distribution

$\pi(\boldsymbol{\Theta})$ on the parameters $\boldsymbol{\Theta} = \boldsymbol{\mathcal{G}}(\boldsymbol{\theta}, \boldsymbol{\phi}, \boldsymbol{\eta})$. Given noisy observations $\boldsymbol{Y}_{0:M}$ from (2.55), our likelihood function is defined as (2.30). Combining the prior with the likelihood yields the posterior distribution

$$p(\boldsymbol{\Theta} \mid \boldsymbol{Y}_{0:M}) \propto \pi(\boldsymbol{\Theta}) \times \mathcal{L}(\boldsymbol{\Theta} \mid \boldsymbol{Y}_{0:M}). \tag{2.56}$$

We assume the observation model parameter $\phi = 0.2$ to be known, so the learning problem focuses only on estimating the ODE model parameters $\boldsymbol{\theta} = (a, b, c, V(0), R(0))$ and the Gaussian Markov process tuning parameters $\boldsymbol{\eta} = (\sigma_1, \sigma_2)$. We apply the following parameter transformation,

$$\boldsymbol{\Theta} = \boldsymbol{\mathcal{G}}(\boldsymbol{\theta}, \boldsymbol{\eta}) = \big(\boldsymbol{\Psi}_{\boldsymbol{\theta}} = (\log a, \log b, \log c, V(0), R(0)), \boldsymbol{\Psi}_{\boldsymbol{\eta}} = (\log \sigma_1, \log \sigma_2)\big). \tag{2.57}$$

The prior distribution $\pi(\boldsymbol{\Theta})$ consists of independent Normal$(0, 10^2)$ priors on each element of $\boldsymbol{\Psi}_{\boldsymbol{\theta}}$ while the tuning parameters have a flat prior, $\pi(\boldsymbol{\eta}) \propto 1$.

2.6.1 Parameter Posteriors for the Three Methods

For the Basic method and Fenrir, the high-dimensional latent ODE variables $\boldsymbol{X}_{0:N}$ can be approximately integrated out, giving us the corresponding posterior approximation $\hat{p}(\boldsymbol{\Theta} \mid \boldsymbol{Y}_{0:M})$. While this posterior can be sampled from using MCMC techniques, Bayesian parameter estimation can also be achieved by way of a Bayesian normal approximation (e.g., Gelman et al., 2013, Chapter 4). That is, we approximate the posterior $\hat{p}(\boldsymbol{\Theta} \mid \boldsymbol{Y}_{0:M})$ by a multivariate normal distribution given by

$$\boldsymbol{\Theta} \mid \boldsymbol{Y}_{0:M} \approx \text{Normal}(\hat{\boldsymbol{\Theta}}, \hat{\boldsymbol{V}}), \tag{2.58}$$

where

$$\hat{\boldsymbol{\Theta}} = \arg\max_{\boldsymbol{\Theta}} \log \hat{p}(\boldsymbol{\Theta} \mid \boldsymbol{Y}_{0:M}), \quad \hat{\boldsymbol{V}} = -\left[\frac{\partial^2}{\partial \boldsymbol{\Theta} \partial \boldsymbol{\Theta}^\top} \log \hat{p}(\hat{\boldsymbol{\Theta}} \mid \boldsymbol{Y}_{0:M})\right]^{-1}. \tag{2.59}$$

To obtain posteriors of the parameters on the original scale $(\boldsymbol{\theta}, \boldsymbol{\eta})$, we first sample $\boldsymbol{\Theta}^{(1)}, \ldots, \boldsymbol{\Theta}^{(B)}$ from the normal distribution in (2.58), and then transform the samples back using $(\boldsymbol{\theta}^{(b)}, \boldsymbol{\eta}^{(b)}) = \boldsymbol{\mathcal{G}}^{-1}(\boldsymbol{\Theta}^{(b)})$.

The Bayesian normal approximation has been widely used in Bayesian machine learning applications (MacKay, 1992; Gianniotis, 2019) because it typically requires orders of magnitude fewer evaluations of $\hat{p}(\boldsymbol{\Theta} \mid \boldsymbol{Y}_{0:M})$ than the MCMC alternative. Our implementation of **rodeo** is built using the **JAX** library (Bradbury et al., 2018), ensuring that it

is automatic differentiable and JIT compiled. These features enable fast computations for $\log \hat{p}(\boldsymbol{\Theta} \mid \boldsymbol{Y}_{0:M})$ as well as its gradient and Hessian, which in turn support a wide range of optimization algorithms for $\hat{\boldsymbol{\Theta}}$.

The posterior mode $(\hat{\boldsymbol{\Psi}}_{\boldsymbol{\theta}}, \hat{\boldsymbol{\Psi}}_{\boldsymbol{\eta}})$ is estimated using the Newton-CG optimization algorithm (Nocedal and Wright, 2006), as implemented in the **JAXopt** library (Blondel et al., 2021). To estimate the posterior variance of $\boldsymbol{\Psi}_{\boldsymbol{\theta}}$, we compute the Hessian of $\log \hat{p}(\boldsymbol{\Psi}_{\boldsymbol{\theta}}, \boldsymbol{\Psi}_{\boldsymbol{\eta}} \mid \boldsymbol{Y}_{0:M})$ only with respect to the model parameters $\boldsymbol{\Psi}_{\boldsymbol{\theta}}$ and take its inverse. This approach is justified by the stochastic solver's insensitivity to the values of $\boldsymbol{\eta}$ over several orders of magnitude. Including $\boldsymbol{\Psi}_{\boldsymbol{\eta}}$ in the Hessian calculation results in an over-estimated posterior variance for $\boldsymbol{\Psi}_{\boldsymbol{\theta}}$. This approach has been previously recommended by Tronarp et al. (2022).

For the marginal MCMC method of Chkrebtii et al. (2016), we run Algorithm 2.3 to sample from the joint posterior

$$(\boldsymbol{\Theta}^{(1)}, \boldsymbol{X}_{1:N}^{(1)}), \ldots, (\boldsymbol{\Theta}^{(B)}, \boldsymbol{X}_{1:N}^{(B)}) \sim \hat{p}(\boldsymbol{\Theta}, \boldsymbol{X}_{1:N} \mid \boldsymbol{Y}_{0:M}). \tag{2.60}$$

For the proposal distribution in Algorithm 2.3, we use

$$\boldsymbol{\Theta} \mid \boldsymbol{\Theta}^{(\text{curr})} \sim \text{Normal}(\boldsymbol{\Theta}^{(\text{curr})}, \text{diag}(\boldsymbol{\sigma}_{rw}^2)), \tag{2.61}$$

where $\boldsymbol{\sigma}_{rw}^2 = (\boldsymbol{\sigma}_{rw}^{(1)}, \ldots \boldsymbol{\sigma}_{rw}^{(\dim(\boldsymbol{\Theta}))})$ is a tuning parameter for the MCMC algorithm. We tune this hyperparameter to achieve an acceptance rate between 20% and 30%. We run the Chkrebtii et al. (2016) Algorithm for $100,000$ iterations after burn-in. The marginal posterior $\hat{p}(\boldsymbol{\Theta} \mid \boldsymbol{Y}_{0:M})$ can then be obtained by simply discarding the sampled values of $\boldsymbol{X}_{1:N}$.

For the adapted MAGI method, we use Algorithm 2.4 to compute $p_\beta(\boldsymbol{\Theta}, \tilde{\boldsymbol{U}}_{1:N}, \boldsymbol{Y}_{0:M}, \boldsymbol{Z}_{1:N} = \boldsymbol{0})$. The hyperparameter $\boldsymbol{\eta}$ for the Markov prior is assumed to remain fixed during MCMC. Our experiments indicate an insensitivity to the choice of $\boldsymbol{\eta}$ over orders of magnitude. Thus, for simplicity, we set $\boldsymbol{\eta} = 0.1$. For the β-modulated MAGI posterior in (2.53), we choose the prior temperature $\beta = \boldsymbol{\eta}^{-2} \Delta t^{2-2q} \Delta t'$ where Δt is the discretization step size and $\Delta t'$ is the inter-observation step size. We choose β in this manner so that it is proportional to R_{11}^{-1} from the IBM prior in Section 2.2.1, scaled by the factor $\Delta t'/\Delta t$ as suggested by Yang et al. (2021). To sample from the β-modulated MAGI posterior $p_\beta(\boldsymbol{\Theta}, \tilde{\boldsymbol{U}}_{1:N} \mid \boldsymbol{Y}_{0:M}, \boldsymbol{Z}_{1:N} = \boldsymbol{0})$, we use MCMC algorithms from the Python library **BlackJAX** (Lao and Louf, 2020). Like **rodeo**, **BlackJAX** is built with **JAX** and offers a variety of built-in MCMC algorithms. Here, we use the Hybrid Monte Carlo (HMC) algorithm (Duane et al., 1987) with 200 leapfrog steps. The remaining HMC parameters are tuned using the window adaptation

algorithm (Stan Development Team, 2023). We run HMC for $10,000$ iterations after burn-in.

Figure 2.5 displays the parameter posteriors for the Bayesian normal approximation and MCMC at different solver step sizes $\Delta t = T/N$ for the methods discussed. For a baseline comparison, we also include the posteriors simulated from the Bayesian normal approximation using the Basic method, where the RTS smoother mean is replaced with the solution from the Euler scheme given by

$$\boldsymbol{x}(t + \Delta t) = \boldsymbol{x}(t) + \Delta t \boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{x}(t), t). \tag{2.62}$$

Similarly, the black posterior curves in each subplot of Figure 2.5 correspond to the Basic method, but using a highly accurate deterministic solver, namely RKDP, applied to the FN model (2.54) instead. We shall assume that the output of RKDP is the true ODE solution, and therefore its parameter posteriors as the ground truth. For parameter c, the Euler posteriors do not approximate the true posterior well until $\Delta t = 0.02$. In contrast, the Basic method and Fenrir posteriors closely match the true posterior starting from $\Delta t = 0.1$. The marginal MCMC method covers the true parameter values at $\Delta t \leq 0.05$, while the Markov MAGI MCMC posteriors achieve this at $\Delta t \leq 0.1$.

2.7 Discussion

In this chapter, we make two major contributions to the literature. First, we prove that the convergence rates of the RTS smoother estimates match those of the Kalman filter forward pass. Theorems 2.3 and 2.4 establish that as $N \to \infty$, the RTS smoother mean converges to the true solution, and the variance converges to $\mathbf{0}$. While the same holds for the Kalman filter, the RTS smoother is generally preferred because it provides more accurate estimates (Rauch et al., 1965). Our experiments provide empirical evidence supporting this claim.

Second, we review several parameter inference methods for ODEs and introduce a novel extension of the MAGI method (Yang et al., 2021). Our approach imposes a Gaussian Markov process prior on the solution process, allowing Kalman algorithms to efficiently compute the MAGI posterior with linear time complexity $\mathcal{O}(N)$. We compare these parameter inference methods in a numerical study to highlight the differences between approaches.

While we have empirical evidence suggesting improved convergence rates when increasing from the first-order to the second-order IBM prior, this remains unproven theoretically.

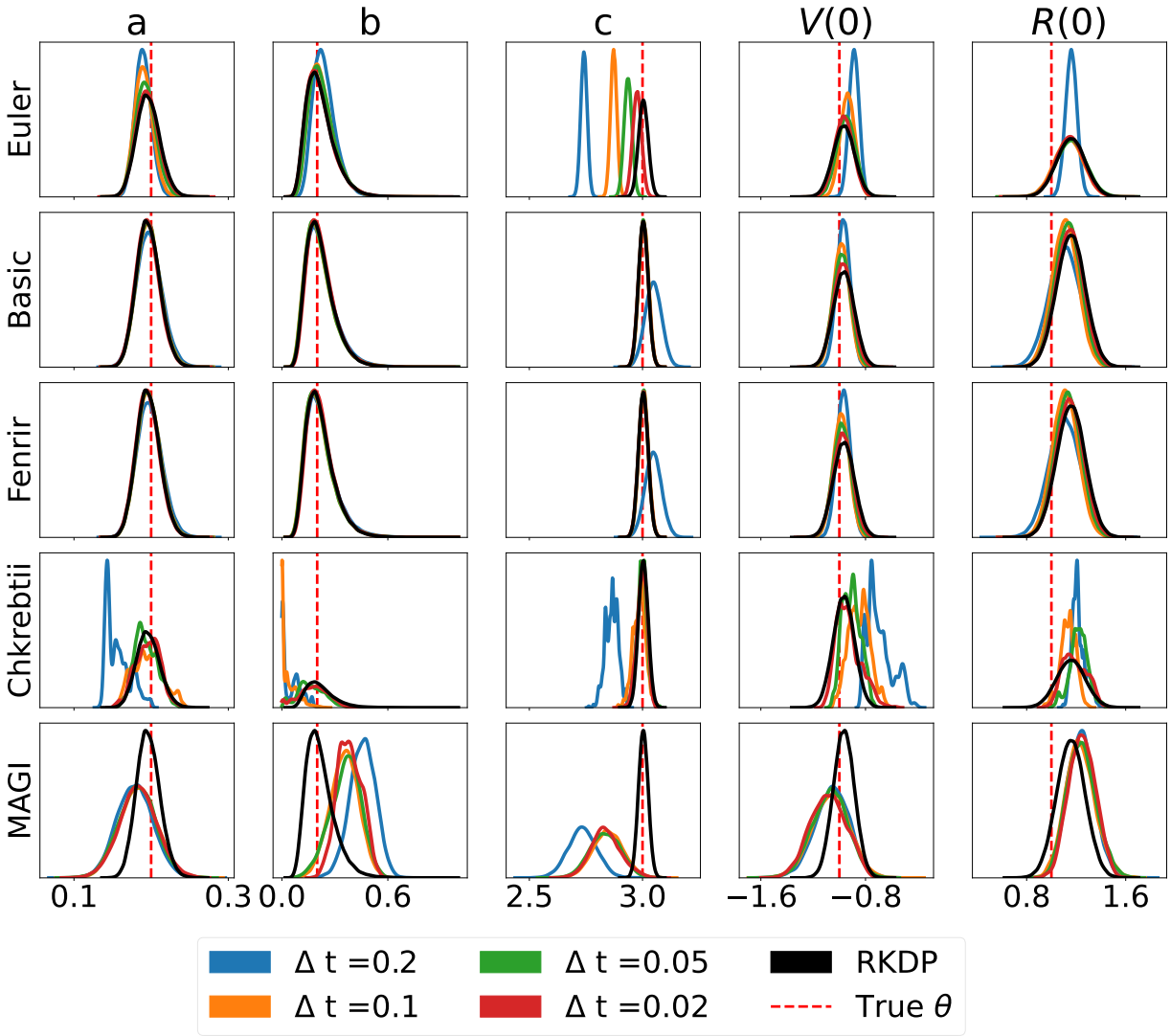

Figure 2.5: Parameter posteriors for the FN model using the Bayesian normal approximation for the Euler ODE solver, the Basic ODE likelihood approximation, and Fenrir. Also included are posteriors from the marginal MCMC method, as well as Markov MAGI sampled via MCMC.

Future research could further investigate the relationship between prior selection and convergence rates for both the Kalman filter and the RTS smoother. Several alternative

interrogation methods have been proposed (Tronarp et al., 2018; Krämer et al., 2022). Two notable approaches are the first-order Taylor expansion and the diagonal Jacobian variation, both of which have shown empirical improvements in ODE solution estimates over the zeroth-order approach (Schober et al., 2019) across multiple experiments (Krämer et al., 2022). However, their theoretical convergence rates remain unproven.

Chapter 3

Data-Adaptive Probabilistic Likelihood Approximation For Ordinary Differential Equations

3.1 Introduction

Parameter estimation for ODEs is a prominent problem in both the machine learning and statistics communities. This process involves solving the ODE repeatedly at different parameter values to compute the likelihood of the observed data. Modern ODE problems generally lack closed-form solutions and, therefore, require computational methods to approximate the solution. Traditionally, this has been done with deterministic algorithms (Diaconis, 1988; Skilling, 1992; Hennig et al., 2015). However, despite these long-established methods, new research in probabilistic numerics shows that probabilistic algorithms provide more reliable estimates in parameter learning problems by directly accounting for numerical uncertainty (Chkrebtii et al., 2016; Conrad et al., 2017).

A standard approach to learning these parameters is to use a parameter search algorithm to find the maximum of the likelihood function. However, many ODEs are extremely sensitive to their model parameters, resulting in numerous sharp local modes in the likelihood function. Consequently, the search algorithm may fail to reach the desired global maximum (Kohberger et al., 1978; Cao et al., 2011; Dass et al., 2017; Rackauckas et al., 2018; Frank et al., 2018). Unlike deterministic solvers, probabilistic ODE solvers can directly use observations to guide the solution process, which can reduce parameter sensitivity (Chkrebtii et al., 2016; Yang et al., 2021; Schmidt et al., 2021; Tronarp et al., 2022).

However, conditioning on observations in a computationally efficient manner adds an extra layer of complexity, and the full potential of probabilistic solvers to reduce parameter sensitivity has yet to be fully realized.

In this chapter, we introduce data-adaptive probabilistic ODE likelihood approximation (DALTON), a novel method in probabilistic numerics designed to address this challenge. More precisely, DALTON approximates the likelihood function for noisy observations in ODE parameter learning problems and offers several major features:

1. *Scalability*: At the core of DALTON lies the Bayesian filtering paradigm (Tronarp et al., 2018) discussed in Chapter 2. This approach, along with its associated approximation methods (Tronarp et al., 2018; Schober et al., 2019; Krämer et al., 2022) enables DALTON to scale linearly with both the number of discretization points N and the ODE dimensions d.

2. *Robustness*: DALTON is able to overcome deep local modes in the ODE likelihood function that both existing probabilistic and deterministic solvers struggle with. This is demonstrated by its insensitivity to the choice of initial values in standard mode-finding algorithms for Bayesian parameter estimation in chaotic ODE systems. Notably, DALTON proves to be more reliable than even direct optimization of the true ODE likelihood, offering improved robustness in challenging inference settings.

3. *Flexibility*: DALTON can be applied to ODE systems with partially observed components and non-Gaussian observation models. For non-Gaussian observations, we present an approximation that enables us to continue using the Bayesian filtering paradigm, thereby maintaining linear scalability.

The remainder of this chapter is structured as follows. Section 3.2 describes the methodology of the DALTON algorithm in detail for both Gaussian and non-Gaussian noisy data. Section 3.3 offers empirical evidence for the three previously mentioned features. Finally, in Section 3.4 we provide some concluding remarks.

3.2 Data-Adaptive Probabilistic ODE Likelihood Approximation

Recall the parameter-dependent IVP introduced in Chapter 2,

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{x}(t), t), \qquad \boldsymbol{x}(0) = \boldsymbol{v}_{\boldsymbol{\theta}}, \qquad t \in [0, T], \tag{3.1}$$

and the surrogate model baeed on the Bayesian filtering paradigm of Tronarp et al. (2018), used to approximate it,

$$\begin{aligned} \boldsymbol{X}_{n+1} \mid \boldsymbol{X}_n &\sim \text{Normal}(\boldsymbol{Q}_{\boldsymbol{\eta}} \boldsymbol{X}_n, \boldsymbol{R}_{\boldsymbol{\eta}}) \\ \boldsymbol{Z}_{n+1} &\overset{\text{ind}}{\sim} \text{Normal}(\dot{\boldsymbol{x}}_{n+1} + \boldsymbol{B}_{n+1} \boldsymbol{x}_{n+1} + \boldsymbol{a}_{n+1}, \boldsymbol{V}_{n+1}). \end{aligned} \tag{3.2}$$

The learning problem consists of estimating the unknown model parameters $\boldsymbol{\theta}$, which determine $\boldsymbol{x}(t)$ in (3.1) from noisy observations $\boldsymbol{Y}_{0:M} = (\boldsymbol{Y}_0, \ldots, \boldsymbol{Y}_M)$, recorded at times $t = t'_0, \ldots, t'_M$ under the observation model

$$\boldsymbol{Y}_i \overset{\text{ind}}{\sim} p(\boldsymbol{Y}_i \mid \boldsymbol{x}(t'_i), \boldsymbol{\phi}), \tag{3.3}$$

with possibly unknown noise parameters $\boldsymbol{\phi}$. Typically, the discretization time points $t = t_0, \ldots, t_N$ used by the ODE solver are much finer than the observation time points. Thus, we define the mapping $n(\cdot)$ such that $t_{n(i)} = t'_i$, and the inverse mapping $i(n) = \max\{i : n(i) \leq n\}$.

The Bayesian filtering model (2.3) from Chapter 2 is then augmented to account for noisy observations from (3.3), as follows (e.g., Schmidt et al., 2021)

$$\begin{aligned} \boldsymbol{X}_{n+1} \mid \boldsymbol{X}_n &\sim \text{Normal}(\boldsymbol{Q}_{\boldsymbol{\eta}} \boldsymbol{X}_n, \boldsymbol{R}_{\boldsymbol{\eta}}) \\ \boldsymbol{Z}_{n+1} &\overset{\text{ind}}{\sim} \text{Normal}(\dot{\boldsymbol{x}}_{n+1} - \boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{x}_{n+1}, t_{n+1}), \boldsymbol{V}) \\ \boldsymbol{Y}_i &\overset{\text{ind}}{\sim} p(\boldsymbol{Y}_i \mid \boldsymbol{x}_{n(i)}, \boldsymbol{\phi}), \end{aligned} \tag{3.4}$$

where $\boldsymbol{\eta}$ are tuning parameters of the Gaussian Markov process prior. The likelihood function induced by the probabilistic solver corresponding to (3.4) for all parameters $\boldsymbol{\Theta} = (\boldsymbol{\theta}, \boldsymbol{\phi}, \boldsymbol{\eta})$ is given by (e.g., Kersting et al., 2020b; Tronarp et al., 2022)

$$\mathcal{L}(\boldsymbol{\Theta} \mid \boldsymbol{Y}_{0:M}) = p(\boldsymbol{Y}_{0:M} \mid \boldsymbol{Z}_{1:N} = \boldsymbol{0}, \boldsymbol{\Theta}). \tag{3.5}$$

As discussed in Chapter 2, Fenrir is an efficient algorithm for approximating the likelihood function (3.5). However, its first step, which involves model interrogations, is entirely data-free, meaning it does not incorporate noisy observations $\boldsymbol{Y}_{0:M}$ in the ODE solver. This limitation motivates our data-adaptive approach, DALTON, which introduces two approximations depending on whether the observation model (3.3) follows a Gaussian distribution or not.

3.2.1 Gaussian Observation Model

We first consider the simpler case where the observation model (3.3) follows a Gaussian distribution

$$\boldsymbol{Y}_i \overset{\text{ind}}{\sim} \text{Normal}(\boldsymbol{D}_i^{(\phi)} \boldsymbol{X}_{n(i)}, \boldsymbol{\Omega}_i^{(\phi)}). \tag{3.6}$$

DALTON begins by applying the conditional probability identity to the likelihood function (3.5), yielding

$$p(\boldsymbol{Y}_{0:M} \mid \boldsymbol{Z}_{1:N} = \boldsymbol{0}, \boldsymbol{\Theta}) = \frac{p(\boldsymbol{Y}_{0:M}, \boldsymbol{Z}_{1:N} = \boldsymbol{0} \mid \boldsymbol{\Theta})}{p(\boldsymbol{Z}_{1:N} = \boldsymbol{0} \mid \boldsymbol{\Theta})}. \tag{3.7}$$

On the right-hand side, the denominator $p(\boldsymbol{Z}_{1:N} = \boldsymbol{0} \mid \boldsymbol{\Theta})$ can be computed using the prediction error decomposition (Schweppe, 1965) with the data-free surrogate model (3.2),

$$p(\boldsymbol{Z}_{1:N} \mid \boldsymbol{\Theta}) \approx \prod_{n=1}^{N} \text{Normal}(\boldsymbol{Z}_n; \boldsymbol{\mu}_n^{\boldsymbol{Z}}, \boldsymbol{\Sigma}_n^{\boldsymbol{Z}}). \tag{3.8}$$

The terms $\boldsymbol{\mu}_n^{\boldsymbol{Z}}$ and $\boldsymbol{\Sigma}_n^{\boldsymbol{Z}}$ are given by

$$\begin{aligned} \boldsymbol{\mu}_n^{\boldsymbol{Z}} &= (\boldsymbol{W}^{(1)} + \boldsymbol{B}_n \boldsymbol{W}^{(0)}) \boldsymbol{\mu}_{n|n-1} + \boldsymbol{a}_n \\ \boldsymbol{\Sigma}_n^{\boldsymbol{Z}} &= (\boldsymbol{W}^{(1)} + \boldsymbol{B}_n \boldsymbol{W}^{(0)}) \boldsymbol{\Sigma}_{n|n-1} (\boldsymbol{W}^{(1)} + \boldsymbol{B}_n \boldsymbol{W}^{(0)})^\top + \boldsymbol{V}_n \end{aligned} \tag{3.9}$$

where $\boldsymbol{W}^{(1)} = \begin{bmatrix} 0 & 1 \end{bmatrix}$ and $\boldsymbol{W}^{(0)} = \begin{bmatrix} 1 & 0 \end{bmatrix}$. The numerator can also be computed using the prediction error decomposition, but two cases must be considered:

1. At time points $t_n \notin \{t_{n(0)}, \ldots, t_{n(M)}\}$, where no observations are recorded, we apply the data-free surrogate model (3.2).

2. At time points $t_n = t_{n(i)}$, where observations are recorded, we augment the surrogate model (3.2) with $\boldsymbol{Y}_i$ as

$$\begin{bmatrix} \boldsymbol{Z}_{n(i)} \\ \boldsymbol{Y}_i \end{bmatrix} \overset{\text{ind}}{\sim} \text{Normal}\left(\begin{bmatrix} \dot{\boldsymbol{x}}_{n(i)} + \boldsymbol{B}_{n(i)} \boldsymbol{x}_{n(i)} + \boldsymbol{a}_{n(i)} \\ \boldsymbol{D}_i^{(\phi)} \boldsymbol{X}_{n(i)} \end{bmatrix}, \begin{bmatrix} \boldsymbol{V}_{n(i)} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{\Omega}_i^{(\phi)} \end{bmatrix} \right), \tag{3.10}$$

   We compute the data-free model interrogation coefficients $\boldsymbol{a}_{n(i)}$, $\boldsymbol{B}_{n(i)}$, and $\boldsymbol{V}_{n(i)}$ exactly as in Table 2.1.

The pseudo-code to compute the right-hand side of (3.7) is provided in Algorithm 3.1. In particular, the `interrogate`() function on Line 10 of Algorithm 3.1 performs the interrogation method of Krämer et al. (2022) in Table 2.1. Algorithm 3.1 achieves a computational speed similar to Fenrir, as it requires one forward pass for each of $p(\boldsymbol{Z}_{1:N} = \mathbf{0} \mid \boldsymbol{\Theta})$ and $p(\boldsymbol{Y}_{0:M}, \boldsymbol{Z}_{1:N} = \mathbf{0} \mid \boldsymbol{\Theta})$, whereas Fenrir uses both a forward and backward pass. Additionally, DALTON is able to approximate the ODE solution posterior, $p_{\text{lin}}(\boldsymbol{X}_{1:N} \mid \boldsymbol{Y}_{0:M}, \boldsymbol{Z}_{1:N}, \boldsymbol{\Theta})$, by applying a forward and backward pass to the numerator of (3.10).

3.2.2 Non-Gaussian Observation Model

Now, suppose we have the general observation model (3.3). Let $\boldsymbol{s}_{n(i)}$ and $\boldsymbol{u}_{n(i)}$ denote the observed and unobserved components of $\boldsymbol{x}(t)$ at time $t = t_i'$ respectively, such that $\frac{\partial}{\partial \boldsymbol{u}_{n(i)}} p(\boldsymbol{Y}_i \mid \boldsymbol{x}_{n(i)}, \boldsymbol{\phi}) = \mathbf{0}$, and

$$p(\boldsymbol{Y}_i \mid \boldsymbol{x}_{n(i)}, \boldsymbol{\phi}) = \exp\{g_i(\boldsymbol{Y}_i \mid \boldsymbol{s}_{n(i)}, \boldsymbol{\phi})\}. \tag{3.11}$$

Similar to the Gaussian case, our goal remains to compute the likelihood (3.5). However, we now consider a different identity,

$$\begin{aligned} p(\boldsymbol{Y}_{0:M} \mid \boldsymbol{Z}_{1:N} = \mathbf{0}, \boldsymbol{\Theta}) &= \frac{p(\boldsymbol{Y}_{0:M}, \boldsymbol{X}_{1:N} \mid \boldsymbol{Z}_{1:N} = \mathbf{0}, \boldsymbol{\Theta})}{p(\boldsymbol{X}_{1:N} \mid \boldsymbol{Y}_{0:M}, \boldsymbol{Z}_{1:N} = \mathbf{0}, \boldsymbol{\Theta})} \\ &= \frac{p(\boldsymbol{X}_{1:N} \mid \boldsymbol{Z}_{1:N} = \mathbf{0}, \boldsymbol{\theta}, \boldsymbol{\eta})}{p(\boldsymbol{X}_{1:N} \mid \boldsymbol{Y}_{0:M}, \boldsymbol{Z}_{1:N} = \mathbf{0}, \boldsymbol{\Theta})} \times \prod_{i=0}^{M} \exp\{g_i(\boldsymbol{Y}_i \mid \boldsymbol{s}_{n(i)}, \boldsymbol{\phi})\}, \end{aligned} \tag{3.12}$$

where the identity holds for any value of $\boldsymbol{X}_{0:N}$. On the right-hand side of (3.12), the numerator can be further decomposed as

$$p(\boldsymbol{X}_{1:N} \mid \boldsymbol{Z}_{1:N} = \mathbf{0}, \boldsymbol{\theta}, \boldsymbol{\eta}) = p(\boldsymbol{X}_N \mid \boldsymbol{Z}_{1:N}, \boldsymbol{\theta}, \boldsymbol{\eta}) \prod_{n=0}^{N-1} p(\boldsymbol{X}_n \mid \boldsymbol{X}_{n+1}, \boldsymbol{Z}_{1:N}, \boldsymbol{\theta}, \boldsymbol{\eta}), \tag{3.13}$$

which can be approximated by applying the Kalman filter and smoothing algorithms to the data-free surrogate model in (3.2). Additionally, the product term can be directly computed using the general observations model (3.11). The main challenge lies in the denominator, as the observations $\boldsymbol{Y}_{0:M}$ are non-Gaussian.

If the observations were Gaussian, we could compute the denominator using the augmented surrogate model as in (3.10). However, since this is not the case, we instead propose

Algorithm 3.1 DALTON probabilistic ODE likelihood approximation for Gaussian observations.

1: **procedure** `dalton`$(\boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{x}, t), \boldsymbol{v}_{\boldsymbol{\theta}}, \boldsymbol{Q}_{\boldsymbol{\eta}}, \boldsymbol{R}_{\boldsymbol{\eta}}, \boldsymbol{Y}_{0:M}, \boldsymbol{D}_{0:M}^{(\boldsymbol{\phi})}, \boldsymbol{\Omega}_{0:M}^{(\boldsymbol{\phi})})$

2: $\quad \boldsymbol{W}^{(1)} \leftarrow \begin{bmatrix} 0 & 1 \end{bmatrix}, \qquad \boldsymbol{W}^{(0)} \leftarrow \begin{bmatrix} 1 & 0 \end{bmatrix}$

3: $\quad \boldsymbol{\mu}_{0|0}, \Sigma_{0|0} \leftarrow \boldsymbol{v}, \boldsymbol{0}$ ▷ Initialization

4: $\quad \boldsymbol{Z}_{1:N} \leftarrow \boldsymbol{0}$

5: $\quad \ell_z, \ell_{yz} \leftarrow 0, 0$

6: $\quad i \leftarrow 0$ ▷ Used to map t_n to t'_i

7: ▷ Lines 8-13 approximate $\log p(\boldsymbol{Z}_{1:N} = \boldsymbol{0} \mid \boldsymbol{\Theta})$

8: $\quad$ **for** $n = 1 : N$ **do**

9: $\quad\quad \boldsymbol{\mu}_{n|n-1}, \boldsymbol{\Sigma}_{n|n-1} \leftarrow$ `kalman_predict`$(\boldsymbol{\mu}_{n-1|n-1}, \boldsymbol{\Sigma}_{n-1|n-1}, \boldsymbol{Q}_{\boldsymbol{\eta}}, \boldsymbol{R}_{\boldsymbol{\eta}})$

10: $\quad\quad \boldsymbol{a}_n, \boldsymbol{B}_n, \boldsymbol{V}_n \leftarrow$ `interrogate`$(\boldsymbol{\mu}_{n|n-1}, \boldsymbol{\Sigma}_{n|n-1}, \boldsymbol{f}(\boldsymbol{x}, t, \boldsymbol{\theta}))$

11: $\quad\quad \boldsymbol{\mu}_n^{\boldsymbol{Z}}, \boldsymbol{\Sigma}_n^{\boldsymbol{Z}} \leftarrow$ `kalman_forecast`$(\boldsymbol{\mu}_{n|n-1}, \boldsymbol{\Sigma}_{n|n-1}, \boldsymbol{a}_n, \boldsymbol{W}^{(1)} + \boldsymbol{B}_n \boldsymbol{W}^{(0)}, \boldsymbol{V}_n)$

12: $\quad\quad \ell_z \leftarrow \ell_z +$ `normal_logpdf`$(\boldsymbol{Z}_n = \boldsymbol{0}; \boldsymbol{\mu}_n^{\boldsymbol{Z}}, \boldsymbol{\Sigma}_n^{\boldsymbol{Z}})$

13: $\quad\quad \boldsymbol{\mu}_{n|n}, \boldsymbol{\Sigma}_{n|n} \leftarrow$ `kalman_update`$(\boldsymbol{\mu}_{n|n-1}, \boldsymbol{\Sigma}_{n|n-1}, \boldsymbol{Z}_n, \boldsymbol{a}_n, \boldsymbol{W}^{(1)} + \boldsymbol{B}_n \boldsymbol{W}^{(0)}, \boldsymbol{V}_n)$

14: ▷ Lines 15-25 approximate $\log p(\boldsymbol{Y}_{0:M}, \boldsymbol{Z}_{1:N} = \boldsymbol{0} \mid \boldsymbol{\Theta})$

15: $\quad \ell_{yz} \leftarrow$ `normal_logpdf`$(\boldsymbol{Y}_0; \boldsymbol{D}_0^{(\boldsymbol{\phi})} \boldsymbol{v}_{\boldsymbol{\theta}}, \boldsymbol{\Omega}_0^{(\boldsymbol{\phi})})$

16: $\quad$ **for** $n = 1 : N$ **do**

17: $\quad\quad \boldsymbol{\mu}_{n|n-1}, \boldsymbol{\Sigma}_{n|n-1} \leftarrow$ `kalman_predict`$(\boldsymbol{\mu}_{n-1|n-1}, \boldsymbol{\Sigma}_{n-1|n-1}, \boldsymbol{Q}_{\boldsymbol{\eta}}, \boldsymbol{R}_{\boldsymbol{\eta}})$

18: $\quad\quad \boldsymbol{a}_n, \boldsymbol{B}_n, \boldsymbol{V}_n \leftarrow$ `interrogate`$(\boldsymbol{\mu}_{n|n-1}, \boldsymbol{\Sigma}_{n|n-1}, \boldsymbol{f}(\boldsymbol{x}, t_n, \boldsymbol{\theta}))$

19: $\quad\quad$ **if** $t_n = t_{n(i)}$ **then**

20: $\quad\quad\quad \boldsymbol{Z}_n \leftarrow \begin{bmatrix} \boldsymbol{Z}_n \\ \boldsymbol{Y}_i \end{bmatrix}, \qquad \boldsymbol{W} \leftarrow \begin{bmatrix} \boldsymbol{W}^{(1)} \\ \boldsymbol{D}_i^{(\boldsymbol{\phi})} \end{bmatrix}, \qquad \boldsymbol{B}_n \leftarrow \begin{bmatrix} \boldsymbol{B}_n \boldsymbol{W}^{(0)} \\ \boldsymbol{0} \end{bmatrix}$

21: $\quad\quad\quad \boldsymbol{a}_n \leftarrow \begin{bmatrix} \boldsymbol{a}_n \\ \boldsymbol{0} \end{bmatrix}, \qquad \boldsymbol{V}_n \leftarrow \begin{bmatrix} \boldsymbol{V}_n & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{\Omega}_i^{(\boldsymbol{\phi})} \end{bmatrix}$

22: $\quad\quad\quad i \leftarrow i + 1$

23: $\quad\quad \boldsymbol{\mu}_n^{\boldsymbol{Z}}, \boldsymbol{\Sigma}_n^{\boldsymbol{Z}} \leftarrow$ `kalman_forecast`$(\boldsymbol{\mu}_{n|n-1}, \boldsymbol{\Sigma}_{n|n-1}, \boldsymbol{a}_n, \boldsymbol{W} + \boldsymbol{B}_n, \boldsymbol{V}_n)$

24: $\quad\quad \ell_{yz} \leftarrow \ell_{yz} +$ `normal_logpdf`$(\boldsymbol{Z}_n; \boldsymbol{\mu}_n^{\boldsymbol{Z}}, \boldsymbol{\Sigma}_n^{\boldsymbol{Z}})$

25: $\quad\quad \boldsymbol{\mu}_{n|n}, \boldsymbol{\Sigma}_{n|n} \leftarrow$ `kalman_update`$(\boldsymbol{\mu}_{n|n-1}, \boldsymbol{\Sigma}_{n|n-1}, \boldsymbol{Z}_n, \boldsymbol{a}_n, \boldsymbol{W} + \boldsymbol{B}_n, \boldsymbol{V}_n)$

26:

27: $\quad$ **return** $\ell_{yz} - \ell_z$ ▷ Estimate of $\log p(\boldsymbol{Y}_{0:M} \mid \boldsymbol{Z}_{1:N} = \boldsymbol{0}, \boldsymbol{\Theta})$

approximating $p(\boldsymbol{X}_{1:N} \mid \boldsymbol{Y}_{0:M}, \boldsymbol{Z}_{1:N} = \boldsymbol{0}, \boldsymbol{\Theta})$ using a multivariate normal distribution. Con-

sidering the denominator on the log scale, we first apply Bayes' rule, giving

$$\begin{aligned}
&\log p(\boldsymbol{X}_{1:N} \mid \boldsymbol{Y}_{0:M}, \boldsymbol{Z}_{1:N} = \boldsymbol{0}, \boldsymbol{\Theta}) \\
&= \log p(\boldsymbol{X}_{1:N} \mid \boldsymbol{Z}_{1:N} = \boldsymbol{0}, \boldsymbol{\theta}, \boldsymbol{\eta}) + \sum_{i=0}^{M} g_i(\boldsymbol{Y}_i \mid \boldsymbol{s}_{n(i)}, \boldsymbol{\phi}) + c_1,
\end{aligned} \tag{3.14}$$

where c_1 is a constant with respect to $\boldsymbol{X}_{0:N}$. To simplify notation, we define $h_i(\boldsymbol{s}_{n(i)}) = g_i(\boldsymbol{Y}_i \mid \boldsymbol{s}_{n(i)}, \boldsymbol{\phi})$, making it clear that the gradient and Hessian are taken with respect to $\boldsymbol{s}_{n(i)}$. Taking a second-order Taylor expansion of $h_i(\boldsymbol{s}_{n(i)})$ around $\boldsymbol{s}_{n(i)} = \hat{\boldsymbol{s}}_{n(i)}$, we obtain

$$\begin{aligned}
h_i(\boldsymbol{s}_{n(i)}) &\approx h_i(\hat{\boldsymbol{s}}_{n(i)}) + \nabla h_i(\hat{\boldsymbol{s}}_{n(i)})^\top (\boldsymbol{s}_{n(i)} - \hat{\boldsymbol{s}}_{n(i)}) + \frac{1}{2}(\boldsymbol{s}_{n(i)} - \hat{\boldsymbol{s}}_{n(i)})^\top \nabla^2 h_i(\hat{\boldsymbol{s}}_{n(i)})(\boldsymbol{s}_{n(i)} - \hat{\boldsymbol{s}}_{n(i)}) \\
&= \frac{1}{2}(\boldsymbol{s}_{n(i)} - \hat{\boldsymbol{Y}}_i)^\top \nabla^2 h_i(\hat{\boldsymbol{s}}_{n(i)})(\boldsymbol{s}_{n(i)} - \hat{\boldsymbol{Y}}_i) + c_2
\end{aligned} \tag{3.15}$$

where

1. ∇h_i and $\nabla^2 h_i$ denote the gradient and Hessian of h_i, respectively.
2. $\hat{\boldsymbol{Y}}_i = \hat{\boldsymbol{s}}_{n(i)} - \nabla^2 h_i(\hat{\boldsymbol{s}}_{n(i)})^{-1} \nabla h_i(\hat{\boldsymbol{s}}_{n(i)})$ can be interpreted as pseudo-observations.
3. c_2 is a constant with respect to $\boldsymbol{s}_{n(i)}$.

After substituting the quadratic approximation (3.15) back into (3.14), the right-hand side is entirely quadratic in $\boldsymbol{X}_{0:N}$ since $p(\boldsymbol{X}_{1:N} \mid \boldsymbol{Z}_{1:N} = \boldsymbol{0}, \boldsymbol{\theta}, \boldsymbol{\eta})$ follows a multivariate normal distribution. The only distribution with a quadratic log probability density function is the multivariate normal, meaning we obtain an estimate of $p(\boldsymbol{X}_{1:N} \mid \hat{\boldsymbol{Y}}_{0:M}, \boldsymbol{Z}_{1:N} = \boldsymbol{0}, \boldsymbol{\Theta})$ equivalent to that of a model with Gaussian observations,

$$\begin{aligned}
\boldsymbol{X}_{n+1} \mid \boldsymbol{X}_n &\sim \text{Normal}(\boldsymbol{Q}_{\boldsymbol{\eta}} \boldsymbol{X}_n, \boldsymbol{R}_{\boldsymbol{\eta}}) \\
\boldsymbol{Z}_n &\overset{\text{ind}}{\sim} \text{Normal}(\dot{\boldsymbol{x}}_n - \boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{x}_n, t_n), \boldsymbol{V}) \\
\hat{\boldsymbol{Y}}_i &\overset{\text{ind}}{\sim} \text{Normal}(\boldsymbol{s}_{n(i)}, -[\nabla^2 h_i(\hat{\boldsymbol{s}}_{n(i)})]^{-1}).
\end{aligned} \tag{3.16}$$

We may now linearize (3.16) in the same way as in the Gaussian setting of Section 3.2.1 to obtain an estimate,

$$p_{\text{lin}}(\boldsymbol{X}_{1:N} \mid \hat{\boldsymbol{Y}}_{0:M}, \boldsymbol{Z}_{1:N} = \boldsymbol{0}), \tag{3.17}$$

which can be computed using the Kalman filter and the RTS sampler. However, we have yet to choose $\hat{\boldsymbol{s}}_{n(i)}$ to plug into the Taylor expansion (3.15). A natural choice is

$$\hat{\boldsymbol{s}}_{n(i)} = E_{\text{lin}}[\boldsymbol{s}_{n(i)} \mid \boldsymbol{Z}_{1:n(i)-1}, \hat{\boldsymbol{Y}}_{0:i-1}, \boldsymbol{\Theta}],$$

which represents the predicted mean of the surrogate model obtained from (3.16). Since (3.12) holds for any value of $\boldsymbol{X}_{0:N}$, we set

$$\boldsymbol{X}_{0:N} = E_{\text{lin}}[\boldsymbol{X}_{0:N} \mid \boldsymbol{Z}_{1:N} = \boldsymbol{0}, \hat{\boldsymbol{Y}}_{0:M}, \boldsymbol{\Theta}]$$

for numerical stability.

The pseudo-code to compute the non-Gaussian DALTON in (3.12) is provided in Algorithm B.1 in Appendix B.1. In terms of computational speed, Algorithm B.1 is approximately twice as slow as Algorithm 3.1, since the former requires two forward and two backward passes, whereas the latter uses only two forward passes.

Both Fenrir and DALTON incorporate observations into the Kalman filtering algorithm when approximating the ODE solution posterior or the observation likelihood. The key distinction between them lies in their approach. Fenrir first linearizes the model *before* incorporating observations (linearize-then-observe). In contrast, DALTON first incorporates observations and then linearizes the model *afterwards* (observe-then-linearize). Furthermore, Fenrir is limited to Gaussian observation models, whereas DALTON can be applied to both Gaussian and non-Gaussian models.

In terms of scalability, DALTON scales linearly with both the number of time discretization points, N, and the number of ODE dimensions. The linear scaling in N arises from the Markov structure of the prior on $\boldsymbol{X}(t)$, while the linear scaling in d is achieved through the model interrogation method of Krämer et al. (2022), detailed in Section 2.2.2 of Chapter 2.

3.3 Numerical Experiments on Parameter Inference

In this section, we evaluate the performance of DALTON through a series of numerical experiments on parameter estimation. We adopt a Bayesian perspective by assuming a prior distribution, $\pi(\boldsymbol{\Theta})$, on the parameters $\boldsymbol{\Theta} = (\boldsymbol{\theta}, \boldsymbol{\phi}, \boldsymbol{\eta})$. Given noisy observations $\boldsymbol{Y}_{0:M}$, the likelihood function is defined as (3.5). Combining the prior with the likelihood gives the posterior distribution

$$p(\boldsymbol{\Theta} \mid \boldsymbol{Y}_{0:M}) \propto \pi(\boldsymbol{\Theta}) \times \mathcal{L}(\boldsymbol{\Theta} \mid \boldsymbol{Y}_{0:M}). \tag{3.18}$$

For DALTON, the latent ODE variables $\boldsymbol{X}_{0:N}$ can be approximately integrated out using (3.7) in the Gaussian setting and (3.12) in the non-Gaussian setting, resulting in an approximate likelihood function $\hat{\mathcal{L}}(\boldsymbol{\Theta} \mid \boldsymbol{Y}_{0:M})$, and consequently an approximate posterior $\hat{p}(\boldsymbol{\Theta} \mid \boldsymbol{Y}_{0:M})$.

We then use the Bayesian normal approximation described in Section 2.6.1 of Chapter 2 to approximate $\hat{p}(\boldsymbol{\Theta} \mid \boldsymbol{Y}_{0:M})$ by a multivariate normal distribution,

$$\boldsymbol{\Theta} \mid \boldsymbol{Y}_{0:M} \approx \text{Normal}(\hat{\boldsymbol{\Theta}}, \hat{\boldsymbol{V}}), \tag{3.19}$$

where

$$\hat{\boldsymbol{\Theta}} = \arg\max_{\boldsymbol{\Theta}} \log \hat{p}(\boldsymbol{\Theta} \mid \boldsymbol{Y}_{0:M}), \quad \hat{\boldsymbol{V}} = -\left[\frac{\partial^2}{\partial \boldsymbol{\Theta} \partial \boldsymbol{\Theta}^\top} \log \hat{p}(\hat{\boldsymbol{\Theta}} \mid \boldsymbol{Y}_{0:M})\right]^{-1}. \tag{3.20}$$

We implemented DALTON in Python using the **JAX** (Bradbury et al., 2018) library to enable automatic differentiation (AD) and JIT compilation. These features allow fast computations for $\log \hat{p}(\boldsymbol{\Theta} \mid \boldsymbol{Y}_{0:M})$ as well as its gradient and Hessian, which in turn support a wide range of optimization algorithms for $\hat{\boldsymbol{\Theta}}$.

For each experiment, we simulate data by first solving the IVP with true $\boldsymbol{\theta}$ using a highly accurate deterministic solver, namely, the RKDP (Dormand and Prince, 1980) as implemented in the **diffrax** library (Kidger, 2021). Observation noise is then added to the ODE solution using the default pseudo-random number generator in Python. Further implementation details of the experiment are provided in Appendix B.2.

3.3.1 SEIRAH model

The SEIRAH model is a six-compartment epidemiological model used in previous studies to describe COVID-19 dynamics in China (Tian et al., 2020) and in France (Prague et al., 2020), where it was found to provide a reasonable fit to observed data. It divides the population into six compartments: susceptible (S), latent (E), ascertained infectious (I), removed (R, including recovered and deceased), unascertained infectious (A), and hospitalized (H). The evolution of these compartments over time is governed by the following system of equations,

$$\begin{aligned}
\frac{\mathrm{d}S(t)}{\mathrm{d}t} &= -\frac{bS(t)(I(t)+\alpha A(t))}{N}, & \frac{\mathrm{d}E(t)}{\mathrm{d}t} &= \frac{bS(t)(I(t)+\alpha A(t))}{N} - \frac{E(t)}{D_e}, \\
\frac{\mathrm{d}I(t)}{\mathrm{d}t} &= \frac{rE(t)}{D_e} - \frac{I(t)}{D_q} - \frac{I(t)}{D_I}, & \frac{\mathrm{d}R(t)}{\mathrm{d}t} &= \frac{I(t)+A(t)}{D_I} + \frac{H_t}{D_h}, \\
\frac{\mathrm{d}A(t)}{\mathrm{d}t} &= \frac{(1-r)E(t)}{D_e} - \frac{A(t)}{D_I}, & \frac{\mathrm{d}H(t)}{\mathrm{d}t} &= \frac{I(t)}{D_q} - \frac{H(t)}{D_h},
\end{aligned} \tag{3.21}$$

where $N = S(t) + E(t) + I(t) + R(t) + A(t) + H(t)$ is the total population size, and the parameter D_h is fixed at 30 (Prague et al., 2020).

Unfortunately, observations are only indirectly available for two of the six compartments, represented by

$$\tilde{I}(t) = \frac{rE(t)}{D_e}, \quad \tilde{H}(t) = \frac{I(t)}{D_q} \tag{3.22}$$

which indicate the number of new cases entering the respective compartments. The observation model of Prague et al. (2020) is

$$Y_i^{(K)} \overset{\text{ind}}{\sim} \text{Poisson}(\tilde{K}(t_i)), \qquad K \in \{I, H\}, \tag{3.23}$$

where each time point $t_i = i$ corresponds to days, for $i = 0, \ldots, 60$. The unknown parameters to be estimated in this model are $\boldsymbol{\theta} = (b, r, \alpha, D_e, D_q, E(0), I(0))$. Experimental data were simulated using the true parameters $\boldsymbol{\theta} = (2.23, 0.034, 0.55, 5.1, 2.3, 1.13, 15492, 21752)$ and the remaining initial values $S(0) = 63884630$, $R(0) = 0$, $A(0)_0 = 618013$, and $H(0)_0 = 13388$. These values were reported by Prague et al. (2020) based on the COVID-19 events in France.

Since the observation model is non-Gaussian, Fenrir is not applicable in this example. Instead, we compare our DALTON method with two other ODE solvers. The first is the previously described RKDP, for which we perform a Bayesian normal approximation using the likelihood function

$$\mathcal{L}(\boldsymbol{\Theta} \mid \boldsymbol{Y}_{0:M}) \propto \prod_{i=0}^{M} p(\boldsymbol{Y}_i \mid \boldsymbol{X}_{n(i)} = \boldsymbol{\mu}_{n(i)}(\boldsymbol{\theta}), \boldsymbol{\phi}). \tag{3.24}$$

where $\boldsymbol{\mu}_{n(i)}(\boldsymbol{\theta})$ denotes the true solution of the ODE system obtained via RKDP for given parameters $\boldsymbol{\theta}$. We refer to this as the true likelihood, as it explicitly evaluates the likelihood using the true ODE solution.

The second is the marginal MCMC method of Chkrebtii et al. (2016), described in Section 2.5.3 and Algorithm 2.3. We use a Gaussian random walk as the proposal distribution, with a hyperparameter controlling the proposal standard deviation. This hyperparameter is tuned to achieve an overall acceptance rate of 25% during the burn-in phase. The algorithm is then run for $100,000$ iterations after burn-in.

Figure 3.1 compares the parameter posteriors obtained from DALTON, RKDP, and the marginal MCMC method across various step sizes Δt. Since the observations are non-Gaussian, we use the non-Gaussian variant of DALTON described in Section 3.2.2, specifically implemented via Algorithm B.1. At the largest step size ($\Delta t = 0.2$), DALTON's

posterior intervals cover all the true parameter values, whereas the marginal MCMC intervals only cover parameters r and D_e. As the step size decreases, the DALTON posteriors rapidly converge toward those obtained from RKDP. In contrast, even at the smallest step size ($\Delta t = 0.01$), the marginal MCMC approximation has not fully converged to the RKDP posteriors.

3.3.2 Lorenz63 model

We now examine a chaotic ODE system that is known to be highly sensitive to its model parameters. The Lorenz63 model (Lorenz, 1963) describes the behaviour of a fluid layer heated from below and cooled from above. It is defined by a system of three ODEs for the state vector $\boldsymbol{x}(t) = (x(t), y(t), z(t))$ commonly referred to as the Lorenz equations, which are given by

$$\dot{x}(t) = \alpha(y(t) - x(t)), \quad \dot{y}(t) \quad = x(t)(\rho - z(t)) - y(t), \quad \dot{z}(t) = x(t)y(t) - \beta z(t). \tag{3.25}$$

Our goal is to estimate six parameters $\boldsymbol{\theta} = (\rho, \alpha, \beta, x(0), y(0), z(0))$, where $\rho, \alpha, \beta > 0$. The observation model is given by

$$\boldsymbol{Y}_i \overset{\text{ind}}{\sim} \text{Normal}(\boldsymbol{x}(t_i), \phi^2 \cdot \boldsymbol{I}_{3\times 3}), \tag{3.26}$$

where $t_i = i$ with $i = 0, 1, \ldots 20$ and $\phi^2 = 0.005$. Since the observations are Gaussian, this example allows us to compare the performance of DALTON with Fenrir.

Before addressing the parameter estimation problem, we first evaluate the quality of the ODE solution estimates produced by Fenrir and DALTON for the Lorenz63 model, and compare them to the true solution computed by RKDP using the true parameter values $\boldsymbol{\theta} = (28, 10, 8/3, -12, -5, 38)$. For reference, we also approximate the ODE solution using the data-free probabilistic solver in (2.9). Figure 3.2 displays the resulting solution curves for $x(t)$ starting at time $t = 5$, for each probabilistic method at various step sizes. Between $t = 0$ and $t = 5$, all probabilistic methods yielded nearly identical estimates, so we omit this portion to highlight the system's chaotic behaviour. The solution curves for $y(t)$ and $z(t)$ exhibit similar trends and are therefore omitted for brevity. The complete set of trajectories for all latent states is provided in Figure B.1 in Appendix B.3.

Without guidance from observational data, the data-free probabilistic solver quickly diverges from the true solution, becoming highly inaccurate beyond a certain point. Fenrir also struggles with the chaotic nature of the Lorenz63 system due to its reliance on a linearize-then-observe approach, where observational data is incorporated only after model

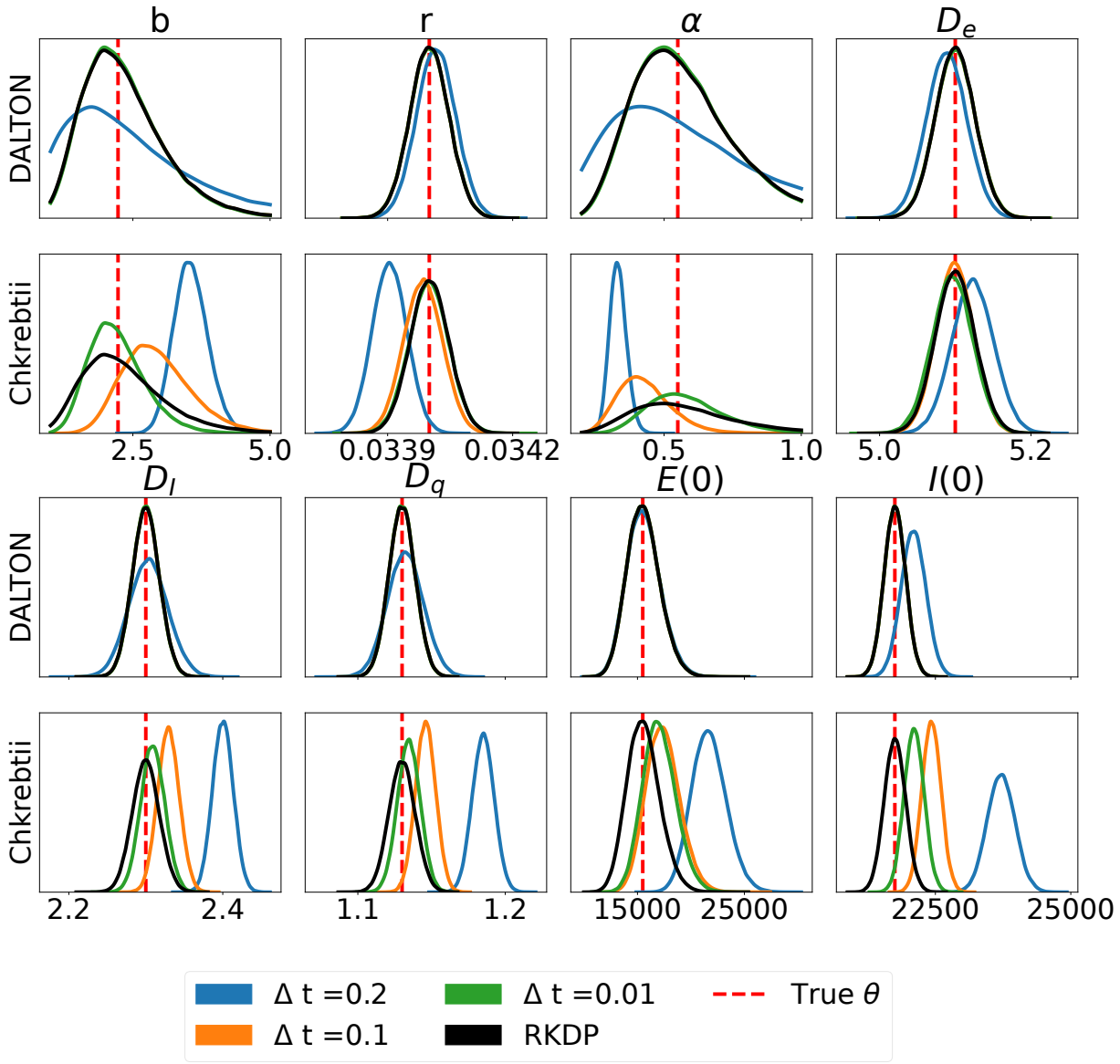

Figure 3.1: Parameter posteriors for the SEIRAH model using various ODE solvers.

linearization. This limitation is evident in Figure 3.2, where the Fenrir solution curves pass through the observed data points, but deviate significantly from the true ODE trajectory elsewhere. In contrast, DALTON, which follows an observe-then-linearize strategy, incor-

porates the observations before linearization, resulting in solution curves that more closely follow the ground truth.

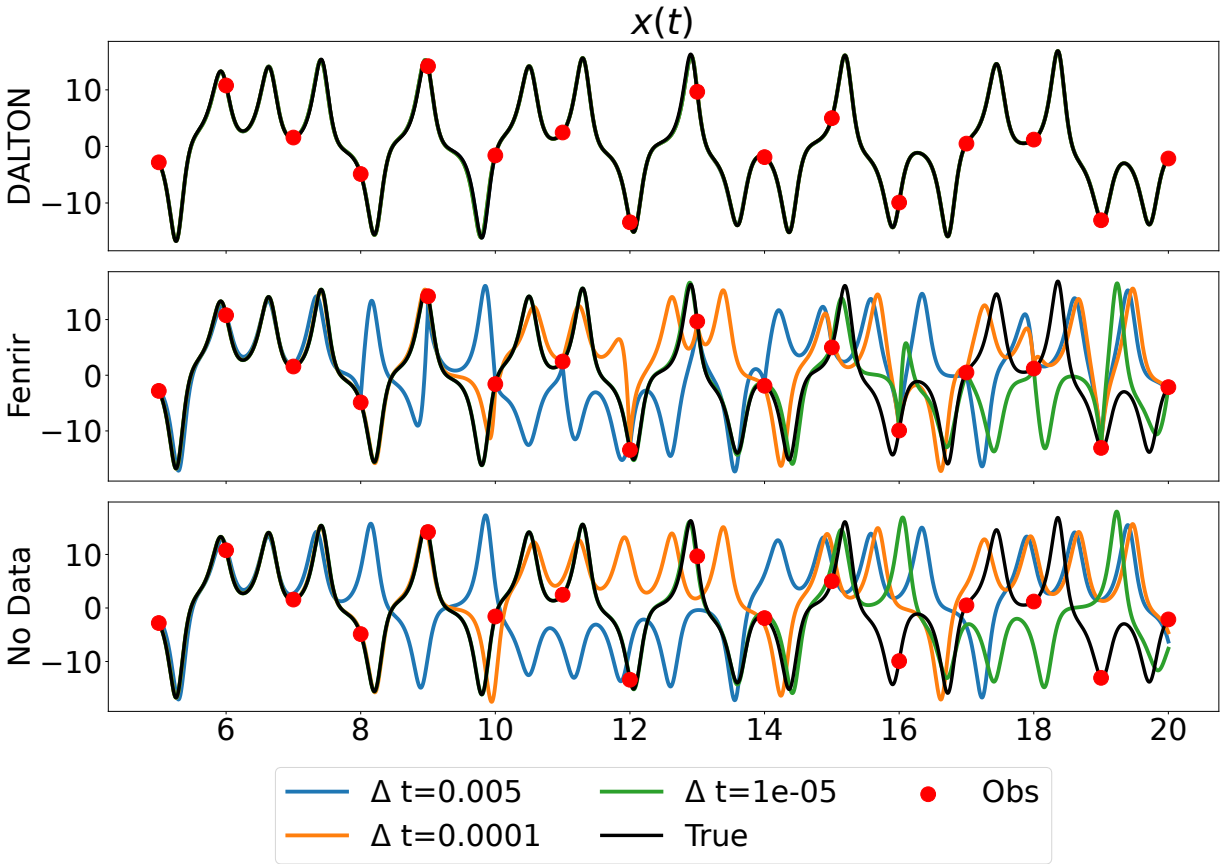

Figure 3.2: $x(t)$ for the Lorenz63 model calculated using various solvers.

Figure 3.3 presents the posterior distributions estimated via the Bayesian normal approximation using DALTON, Fenrir and RKDP, each initialized with two different starting values for $\boldsymbol{\theta}$ in the mode-finding optimization algorithm. The first is set at the true value of $\boldsymbol{\theta}$, and the second is offset by 0.1. The DALTON posteriors consistently cover the true parameter values and converge to the same solution regardless of the initial value. In contrast, the Fenrir and RKDP posteriors only align with the truth when initialized at the correct parameter values. Additionally, both methods yield extremely narrow posterior variances tightly centered around their respective starting points, indicating sensitivity to

initialization. This behaviour suggests that the likelihood surfaces for Fenrir and RKDP contain sharp local optima, which DALTON is able to remove.

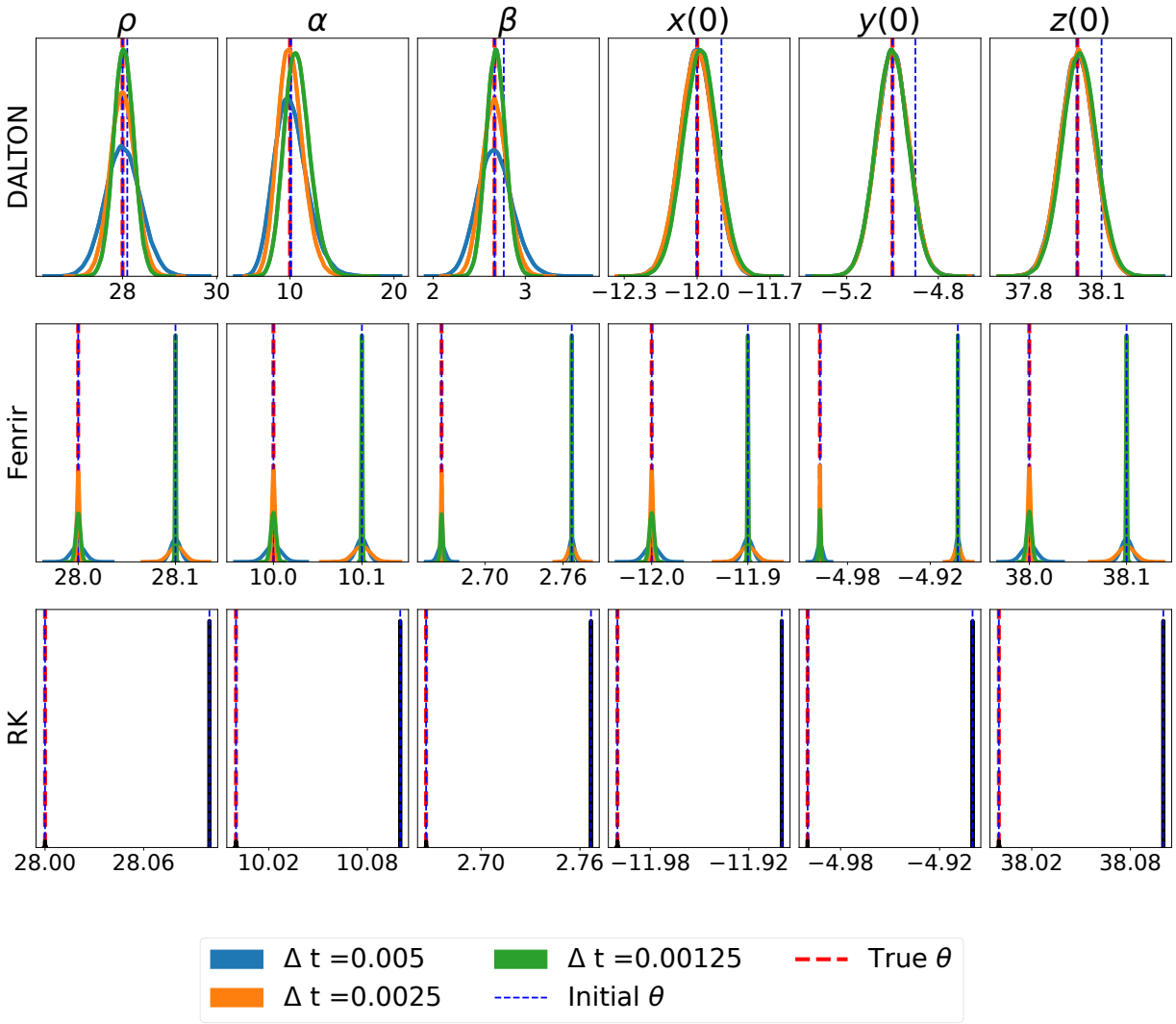

Figure 3.3: Parameter posteriors for the Lorenz63 model using various ODE solvers at two initial parameter values.

To investigate this claim, we conducted an additional experiment to examine how the choice of initial values affects convergence for each of the three methods. We simplified the

parameter estimation problem by assuming that the initial ODE values $(x(0), y(0), z(0))$ are known, and that observations are available at $t_i = i/10$ for $i = 0, \ldots, 200$. This simplified parameter learning experiment is designed to clarify the issue observed in Figure 3.3, where both Fenrir and RKDP failed to move away from their initializations.

Figure 3.4 displays the posterior modes of α, ρ, and β as a function of different initializations in the mode-finding algorithms for DALTON, Fenrir and RKDP. When initialized far from the true values, both Fenrir and RKDP fail to recover the ground truth. In contrast, DALTON consistently identifies the correct parameter values across a broad range of starting points.

As observed earlier in Figure 3.3, Fenrir and RKDP also produce extremely narrow uncertainty estimates for all parameters. This lack of variability, combined with their failure to converge when initialized away from the true values, suggests that mode-finding algorithms applied to their likelihood approximations are sensitive to initialization and can become trapped in local optima. Consequently, the resulting parameter posteriors may fail to cover the ground truth. In contrast, DALTON consistently maintains posterior coverage of the true parameters, regardless of the initialization, indicating greater robustness to local modes during optimization.

Figure 3.5 shows the DALTON solution posterior at $\Delta t = 0.00125$ for component $x(t)$ (the remaining components are displayed in Figure B.2 of Appendix B.3). Care must be taken in computing this posterior, since exact ODEs simulated from the DALTON parameter posteriors in Figure 3.3 look nothing like the true solution, due to extreme parameter sensitivity. Instead, the solution posterior is taken with respect to the probabilistic model (3.4), i.e., each curve in Figure 3.5 is obtained by first drawing $\boldsymbol{\Theta}$ from the DALTON posterior in Figure 3.3, then drawing $x_{0:N}$ from the linearized approximation (3.17) to $p(\boldsymbol{X}_{1:N} \mid \boldsymbol{\Theta}, \boldsymbol{Y}_{0:M}, \boldsymbol{Z}_{1:N} = \boldsymbol{0})$. Figure 3.5 indicates that these solution posteriors closely resemble the true ODE.

These experiments demonstrate that the posteriors produced by DALTON converge to the true posteriors in ODE problems with mild parameter sensitivity. In contrast, for problems with extreme parameter sensitivity, the surrogate SSM used by DALTON allows for greater variability in the ODE approximation than the exact solution, as illustrated in Figure 3.5. This added flexibility reduces the sensitivity of the likelihood to parameter changes, making it easier for optimization algorithms to identify the correct mode. When the true posteriors are desired, DALTON could potentially be used to obtain good starting values from which to initialize the calculation of the Bayesian normal approximation (3.20) with the exact likelihood of RKDP. This seems to work for the data of the Lorzenz63 example of Section 3.3.2, though a more fulsome exploration of this approach is left for

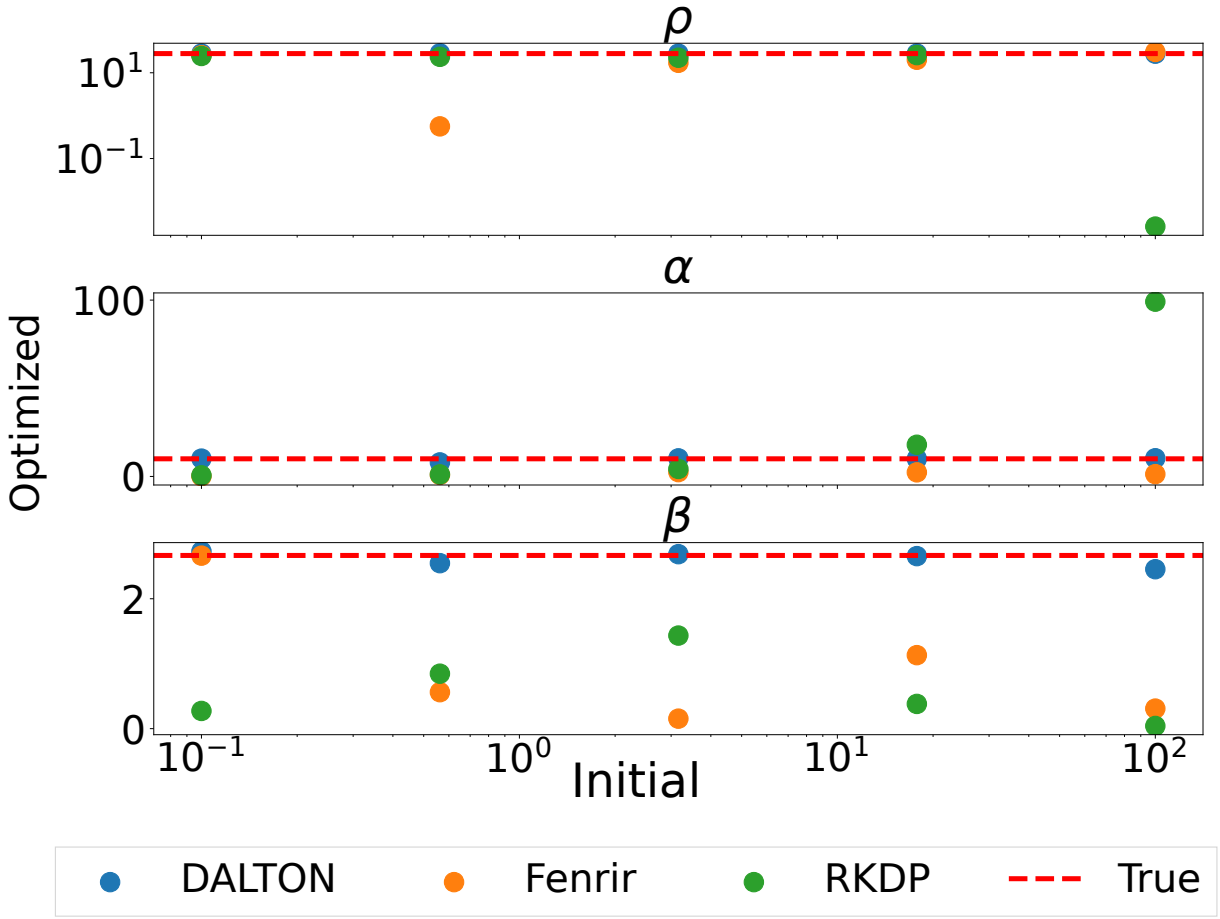

Figure 3.4: Initial and optimized parameter values for the Lorenz63 model using various ODE solvers.

future work.

3.4 Discussion

In this chapter, we introduced DALTON, a data-adaptive probabilistic approximation method for the typically intractable parameter likelihood function of ODEs. DALTON is built upon a well-established Bayesian filtering paradigm, leveraging efficient Kalman

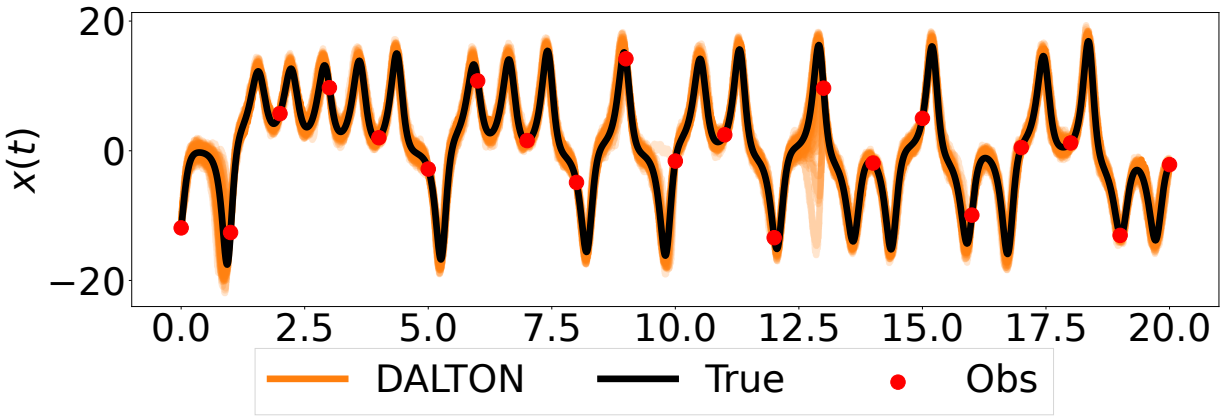

Figure 3.5: ODE posteriors of $x(t)$ for the Lorenz63 model using DALTON with $\Delta t = 0.00125$.

recursions to achieve linear scalability in both the number of ODE dimensions and the number of discretization points. It incorporates information from observed data during the ODE solution process, resulting in more reliable parameter estimates compared to existing probabilistic solvers. Moreover, in cases involving deep local modes, DALTON has proven to be even more robust than using the exact ODE likelihood itself. Finally, DALTON is versatile and applicable to observation models with arbitrary distributions, including those with partially unobserved components.

Despite strong empirical performance, DALTON currently lacks formal theoretical convergence guarantees. Preliminary convergence results for the underlying probabilistic solvers used in both DALTON and Fenrir have been established: the forward pass by Kersting et al. (2020b) and the backward pass in Chapter 2. A promising direction for future work is to establish conditions under which the likelihood approximation of DALTON converges to the true likelihood.

As it currently stands, DALTON uses a fixed step size, requiring manual tuning for optimal performance on a per-problem basis. A promising direction for future research is the adaptive selection of step size, following techniques such as those proposed by Schober et al. (2019). Additionally, like other probabilistic ODE likelihood approximation methods (Yang et al., 2021; Tronarp et al., 2022), DALTON currently lacks theoretical guarantees of convergence. Lastly, exploring the effectiveness of DALTON in parameter inference problems for stiff ODE systems presents an interesting avenue for further investigation.

Chapter 4

An Information-Efficient Variational Inference Method for Stochastic Differential Equations

4.1 Introduction

SDEs are powerful tools for modelling stochastic phenomena across many fields in the natural sciences, finance, and engineering (Särkkä and Solin, 2019). A central problem in working with SDEs is accurately estimating their parameters from noisy or partial observations. The parameter-dependent diffusion process governed by the SDE is given by

$$\mathrm{d}\boldsymbol{X}(t) = \boldsymbol{\alpha}(\boldsymbol{X}(t), \boldsymbol{\theta})\,\mathrm{d}t + \boldsymbol{\beta}(\boldsymbol{X}(t), \boldsymbol{\theta})^{1/2}\,\mathrm{d}\boldsymbol{B}(t), \qquad \boldsymbol{X}(0) = \boldsymbol{v}_{\boldsymbol{\theta}}, \qquad t \in [0, T], \tag{4.1}$$

where $\boldsymbol{X}(t)$ is a d-dimensional random vector, $\boldsymbol{\alpha}$ is a d-dimensional drift vector, $\boldsymbol{\beta}$ is a $d \times d$ positive definite diffusion matrix, and $\boldsymbol{B}(t)$ is a d-dimensional Brownian motion. The notation $\boldsymbol{\beta}^{1/2}$ denotes the matrix square root, such that $\boldsymbol{\beta}^{1/2}(\boldsymbol{\beta}^{1/2})^\top = \boldsymbol{\beta}$. Both the drift and diffusion coefficients depend on the unknown model parameter $\boldsymbol{\theta}$. We assume that $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are sufficiently regular functions to ensure that (4.1) has a weak non-explosive solution (Øksendal, 2003).

Like ODEs, SDEs typically cannot be solved in closed form and must instead be dis-

cretized, leading to a generalized SSM

$$\begin{aligned} \boldsymbol{X}_0 &\sim \pi(\boldsymbol{X}_0 \mid \boldsymbol{\theta}), \\ \boldsymbol{X}_m &\sim p(\boldsymbol{X}_m \mid \boldsymbol{X}_{m-1}, \boldsymbol{\theta}), \\ \boldsymbol{Y}_m &\overset{\text{iid}}{\sim} p(\boldsymbol{Y}_m \mid \boldsymbol{X}(t'_m), \boldsymbol{\theta}), \end{aligned} \tag{4.2}$$

where $\boldsymbol{Y}_m$ is a noisy observation of $\boldsymbol{X}(t'_m)$, with $t'_m = m \cdot \Delta t'$, $\Delta t' = T/M$ and

$$\boldsymbol{X}_m = \{\boldsymbol{X}(t) : t \in \mathcal{T}_m\} \tag{4.3}$$

where $\mathcal{T}_m$ contains all time points in the interval $(t'_{m-1}, t'_m]$. A detailed explanation of the discretization procedure is provided in Section 4.2.1.

The goal of parameter inference for SDEs is to evaluate the marginal likelihood of the observed data under the SSM defined in (4.2)

$$\begin{aligned} \mathcal{L}(\boldsymbol{\theta} \mid \boldsymbol{Y}_{0:M}) &:= p(\boldsymbol{Y}_{0:M} \mid \boldsymbol{\theta}) \\ &= \int p(\boldsymbol{X}_{0:M}, \boldsymbol{Y}_{0:M} \mid \boldsymbol{\theta}) \, \mathrm{d}\boldsymbol{X}_{0:M} \\ &= \int \left[\pi(\boldsymbol{X}_0 \mid \boldsymbol{\theta}) p(\boldsymbol{Y}_0 \mid \boldsymbol{X}_0, \boldsymbol{\theta}) \prod_{m=1}^{M} p(\boldsymbol{Y}_m \mid \boldsymbol{X}_m, \boldsymbol{\theta}) p(\boldsymbol{X}_m \mid \boldsymbol{X}_{m-1}, \boldsymbol{\theta}) \right] \mathrm{d}\boldsymbol{X}_{0:M} \,, \end{aligned} \tag{4.4}$$

where $\boldsymbol{X}_{0:M} = (\boldsymbol{X}_0, \ldots, \boldsymbol{X}_M)$. Alternatively, one may be interested in the posterior distribution over $\boldsymbol{\theta}$ induced by this likelihood

$$p(\boldsymbol{\theta} \mid \boldsymbol{Y}_{0:M}) \propto p(\boldsymbol{\theta}) \mathcal{L}(\boldsymbol{\theta} \mid \boldsymbol{Y}_{0:M}), \tag{4.5}$$

where $p(\boldsymbol{\theta})$ is a prior. In practice, this requires evaluating the likelihood $\mathcal{L}(\boldsymbol{\theta} \mid \boldsymbol{Y}_{0:M})$ for every $\boldsymbol{\theta}$. However, computing the integral in (4.4) is typically infeasible due to the high dimensionality and complexity of the latent space.

To circumvent this, MCMC methods (Roberts and Stramer, 2001; Golightly and Wilkinson, 2008; Kou et al., 2012; Whitaker et al., 2017) have been widely used to sample from the joint posterior

$$p(\boldsymbol{\theta}, \boldsymbol{X}_{0:M} \mid \boldsymbol{Y}_{0:M}) \propto p(\boldsymbol{\theta}) p(\boldsymbol{X}_{0:M}, \boldsymbol{Y}_{0:M} \mid \boldsymbol{\theta}). \tag{4.6}$$

While this avoids the need to integrate out $\boldsymbol{X}_{0:M}$ directly, it introduces scalability challenges as the effective parameter space grows with the number of latent states, which can be prohibitively large in practice. Particle filtering, in contrast, directly approximates the posterior distribution in (4.5) (Andrieu and Doucet, 2002; Kantas et al., 2009, 2015).

However, these methods also face scalability limitations, particularly in high-dimensional or long time-series settings (Bengtsson et al., 2008; Archer et al., 2015).

A more recent line of research focuses on variational inference, where the goal is to select a family of tractable distributions $q(\boldsymbol{\theta}, \boldsymbol{X}_{0:M}; \boldsymbol{\phi})$ that can be efficiently sampled for any variational parameter $\boldsymbol{\phi}$, and then optimize $\boldsymbol{\phi}$ to closely approximate the intractable posterior $p(\boldsymbol{\theta}, \boldsymbol{X}_{0:M} \mid \boldsymbol{Y}_{0:M})$. The exact formulation is provided in Section 4.2.5. This approach converts the original integration problem in (4.4) into an optimization problem. Thanks to recent advancements in stochastic gradient-based optimization, variational inference offers a substantially faster alternative to MCMC, often by several orders of magnitude.

Several authors have considered variational inference for general SSMs (Archer et al., 2015; Krishnan et al., 2017; Becker et al., 2019; Ruhe and Forré, 2022; Revach et al., 2022) and for SDEs in particular (Ryder et al., 2018; Tzen and Raginsky, 2019; Ghosh et al., 2022). A key requirement for accurate variational inference is that the variational distribution follows the same conditional dependence structure as the true posterior. For SSMs, this corresponds to the factorization

$$\begin{aligned} q(\boldsymbol{\theta}, \boldsymbol{X}_{0:M}; \boldsymbol{\phi}) &= q(\boldsymbol{\theta}; \boldsymbol{\phi}) q(\boldsymbol{X}_{0:M} \mid \boldsymbol{\theta}; \boldsymbol{\phi}) \\ &= q(\boldsymbol{\theta}; \boldsymbol{\phi}) q(\boldsymbol{X}_0 \mid \boldsymbol{\theta}; \boldsymbol{\phi}) \prod_{m=1}^{M} q(\boldsymbol{X}_m \mid \boldsymbol{X}_{m-1}, \boldsymbol{\theta}; \boldsymbol{\phi}), \end{aligned} \tag{4.7}$$

since, conditioned on the observed variables and model parameters, the latent states in a SSM satisfy the Markov property. Many of the variational distributions mentioned above have this structure, achieved with the help of a RNN (Krishnan et al., 2017; Ryder et al., 2018; Becker et al., 2019; Ruhe and Forré, 2022; Ghosh et al., 2022). However, most existing variational methods effectively learn the marginal distribution $q(\boldsymbol{X}_m \mid \boldsymbol{\theta}; \boldsymbol{\phi})$ from the filtering distribution $p(\boldsymbol{X}_m \mid \boldsymbol{Y}_{0:m}, \boldsymbol{\theta})$. In contrast, the smoothing distribution $p(\boldsymbol{X}_m \mid \boldsymbol{Y}_{0:M}, \boldsymbol{\theta})$ leverages all observations when estimating each latent state, making more efficient use of available information and typically yielding more accurate results than filtering alone (Rauch et al., 1965; Doucet and Johansen, 2009). Nevertheless, scalable variational approximations to the smoothing distribution remain limited.

To the best of our knowledge, we present the first information-efficient variational inference (IEVI) method for SSMs–and SDEs in particular–that captures not only the correct conditional dependence structure, but also the correct *information flow* from the observed variables $\boldsymbol{Y}_{0:M}$ to the posterior distribution $p(\boldsymbol{X}_{0:M} \mid \boldsymbol{\theta}, \boldsymbol{Y}_{0:M})$. This is accomplished by constructing a RNN for $q(\boldsymbol{X}_{0:M} \mid \boldsymbol{\theta}; \boldsymbol{\phi})$ that mimics the forward-backward recursions used to compute the smoothing distribution of SSMs (Doucet and Johansen, 2009).

A major advantage of our approach is that our variational model for the conditional latent variable distribution is actually of the form

$$q(\boldsymbol{X}_{0:M} \mid \boldsymbol{\theta}, \boldsymbol{Y}_{0:M}; \boldsymbol{\phi}) \tag{4.8}$$

with the same $\boldsymbol{\phi}$ providing the variational approximation for any $\boldsymbol{\theta}$, $\boldsymbol{Y}_{0:M}$ *and* remarkably, any dataset size M. The upshot of this approach is that training scales sublinearly with M when M becomes sufficiently large and linearly otherwise. For the same reason, IEVI also scales efficiently to SSMs with mixed effects as the number of individual datasets grows.

The remainder of this chapter is structured as follows. In Section 4.2, we review background material on SDEs, existing non-variational methods for approximating the likelihood in (4.4) and the variational inference framework. Section 4.3 introduces the IEVI algorithm. Section 4.4 provides a detailed comparison between IEVI and two closely related works. Section 4.5 presents numerical experiments comparing several parameter inference methods for SDEs, including a large mixed-effects model. Finally, Section 4.6 concludes the chapter.

4.2 Background

4.2.1 Stochastic Differential Equations

The SDE parameter inference problem is to estimate unknown parameter $\boldsymbol{\theta}$ from partially observed, noisy measurements $\boldsymbol{Y}_{0:M} = (\boldsymbol{Y}_0, \ldots, \boldsymbol{Y}_M)$ of the continuous process $\boldsymbol{X}(t)$ governed by (4.1), collected at discrete time points $t = t'_0, \ldots, t'_M$. For ease of presentation, we assume that the observations are evenly spaced, such that $t'_m = m \cdot \Delta t'$, where $\Delta t' = T/M$ denotes the inter-observation step size.

In general, there is no closed-form solution to the SDE in (4.1). It is, therefore, standard practice to approximate the solution numerically over a discretized time grid. This approximation error can be controlled by refining the step size, allowing the numerical solution to converge to the true solution as the step size decreases (Ryder et al., 2018). Often, a smaller step size than the inter-observation step size is required to obtain a sufficiently accurate numerical approximation.

To this end, we approximate the SDE using a step size $\Delta t = \Delta t'/r$ where we refer to $r \geq 1$ as the resolution number. Referring back to the SSM in (4.2), this discretization implies that

$$\boldsymbol{X}_m = (\boldsymbol{X}(((m-1)r+1) \cdot \Delta t), \ldots, \boldsymbol{X}(mr \cdot \Delta t)) \,. \tag{4.9}$$

Perhaps, the simplest discretization scheme is the Euler-Maruyama method (Maruyama, 1955), which approximates the latent states in (4.9) by

$$\boldsymbol{X}(n{\cdot}\Delta t) = \boldsymbol{X}((n{-}1){\cdot}\Delta t){+}\boldsymbol{\alpha}(\boldsymbol{X}((n{-}1){\cdot}\Delta t), \boldsymbol{\theta})\Delta t{+}\boldsymbol{\beta}(\boldsymbol{X}((n{-}1){\cdot}\Delta t), \boldsymbol{\theta})^{1/2}\sqrt{\Delta t}\boldsymbol{z}_n, \quad (4.10)$$

where $\boldsymbol{z}_n$ are independent standard normal vectors for $n = (m-1)r+1, \ldots, mr$.

In the following sections, we describe several non-variational methods that have traditionally been used to approximate the likelihood in (4.4) assuming the SSM in (4.2).

4.2.2 Particle Filtering

The first non-variational method we consider is particle filtering (Doucet and Johansen, 2009; Murray, 2013). The core assumption of particle filtering is that, although it is possible to evaluate the density of any of the distributions in (4.2), we cannot sample from them directly. Instead, we choose proposal distributions $q(\boldsymbol{X}_0 \mid \boldsymbol{Y}_0, \boldsymbol{\theta})$ for the initial time point and $q(\boldsymbol{X}_m \mid \boldsymbol{X}_{m-1}, \boldsymbol{Y}_m, \boldsymbol{\theta})$ for subsequent steps.

Algorithm 4.1 presents a basic particle filtering which approximates the likelihood in (4.4). On line 7, the function `resampler()` transforms a set of weighted particles into a set of unweighted ones that represent the same distribution. Several resampling strategies have been proposed (Douc et al., 2005; Doucet and Johansen, 2009). The most common one is a multinomial resampler, which resamples particles $\boldsymbol{X}_m^{(1:P)}$ with replacement according to the weights $w_m^{(1:P)}$ at time step m. However, the resulting likelihood approximation $\hat{\mathcal{L}}(\boldsymbol{\theta} \mid \boldsymbol{Y}_{0:M})$ is not fully differentiable (Corenflos et al., 2021). While it is possible to compute a gradient estimate by omitting the non-differentiable terms introduced by the multinomial resampler, doing so yields a gradient with a non-vanishing bias that fails to propagate through time (Corenflos et al., 2021). Alternatively, unbiased gradient estimates exist, but they come at a high computational cost of $\mathcal{O}(P^2)$ (Poyiadjis et al., 2011; Corenflos et al., 2021) or suffer from high variance (Karkus et al., 2018; Tan et al., 2024).

We instead use a multivariate normal resampler, which fits a multivariate normal distribution to the weighted particles by computing their weighted mean and variance, and then resamples P new particles from this distribution. This approach results in a differentiable particle filter, enabling gradient-based optimization to efficiently compute the mode and associated variance of the approximate posterior $\hat{p}(\boldsymbol{\theta} \mid \boldsymbol{Y}_{0:M})$ using the Bayesian normal approximation described in Chapter 3. Our experiments indicate that this approximation produces results very similar to those obtained via exact inference methods, as presented

Algorithm 4.1 Particle filtering for estimating the approximate likelihood $\hat{\mathcal{L}}(\boldsymbol{\theta} \mid \boldsymbol{Y}_{0:M})$.

procedure `particle_filter`$(\boldsymbol{Y}_{0:M}, \boldsymbol{\theta})$
  for $i = 1 : P$ **do** ▷ Importance sampler for initial step.
    $\boldsymbol{X}_0^{(i)} \overset{\text{iid}}{\sim} q(\boldsymbol{X}_0 \mid \boldsymbol{Y}_0, \boldsymbol{\theta})$
    $w_0^{(i)} \leftarrow \dfrac{p(\boldsymbol{Y}_0 \mid \boldsymbol{X}_0^{(i)}, \boldsymbol{\theta})\pi(\boldsymbol{X}_0^{(i)} \mid \boldsymbol{\theta})}{q(\boldsymbol{X}_0^{(i)} \mid \boldsymbol{Y}_0, \boldsymbol{\theta})}$

  for $m = 1 : M$ **do**
    $\tilde{\boldsymbol{X}}_{m-1}^{(1:P)} \leftarrow$ `resampler`$(\boldsymbol{X}_{m-1}^{(1:P)}, w_{m-1}^{(1:P)})$ ▷ Resampling step.
    for $i = 1 : P$ **do**
      $\boldsymbol{X}_m^{(i)} \overset{\text{iid}}{\sim} q(\boldsymbol{X}_m \mid \tilde{\boldsymbol{X}}_{m-1}^{(i)}, \boldsymbol{Y}_m, \boldsymbol{\theta})$ ▷ Propagation step.
      $w_m^{(i)} \leftarrow \dfrac{p(\boldsymbol{Y}_m \mid \boldsymbol{X}(t'_m)^{(i)}, \boldsymbol{\theta})p(\boldsymbol{X}_m^{(i)} \mid \tilde{\boldsymbol{X}}_{m-1}^{(i)}, \boldsymbol{\theta})}{q(\boldsymbol{X}_m^{(i)} \mid \tilde{\boldsymbol{X}}_{m-1}^{(i)}, \boldsymbol{Y}_m, \boldsymbol{\theta})}$ ▷ Weighting step.
  $\hat{\mathcal{L}}(\boldsymbol{\theta} \mid \boldsymbol{Y}_{0:M}) \leftarrow \prod_{m=0}^{M} \left(\frac{1}{P} \sum_{i=1}^{P} w_m^{(i)} \right)$

  return $\hat{\mathcal{L}}(\boldsymbol{\theta} \mid \boldsymbol{Y}_{0:M})$

in Section 4.5. We use the **JAX**-based particle filtering implementation available in the **pfjax** library (Lysy et al., 2025) to compute this approximation.

4.2.3 Laplace Approximation

The second non-variational method is the Laplace approximation (Kristensen et al., 2016), which is based on the identity

$$p(\boldsymbol{Y}_{0:M} \mid \boldsymbol{\theta}) = \frac{p(\boldsymbol{X}_{0:M}, \boldsymbol{Y}_{0:M} \mid \boldsymbol{\theta})}{p(\boldsymbol{X}_{0:M} \mid \boldsymbol{Y}_{0:M}, \boldsymbol{\theta})}. \tag{4.11}$$

The key idea is to approximate the intractable posterior $p(\boldsymbol{X}_{0:M} \mid \boldsymbol{Y}_{0:M}, \boldsymbol{\theta})$ with a multivariate normal given by

$$\boldsymbol{X}_{0:M} \mid \boldsymbol{Y}_{0:M}, \boldsymbol{\theta} \approx \text{Normal}(\hat{\boldsymbol{X}}_{0:M}, \hat{\boldsymbol{V}}_{\boldsymbol{X}}), \tag{4.12}$$

where

$$\begin{aligned} \hat{\boldsymbol{X}}_{0:M} &= \underset{\boldsymbol{X}_{0:M}}{\arg\max} \log p(\boldsymbol{X}_{0:M} \mid \boldsymbol{Y}_{0:M}, \boldsymbol{\theta}), \\ \hat{\boldsymbol{V}}_{\boldsymbol{X}} &= -\left[\frac{\partial^2}{\partial \boldsymbol{X}_{0:M} \partial \boldsymbol{X}'_{0:M}} \log p(\hat{\boldsymbol{X}}_{0:M} \mid \boldsymbol{Y}_{0:M}, \boldsymbol{\theta}) \right]^{-1}. \end{aligned} \tag{4.13}$$

The approximate likelihood $\hat{p}(\boldsymbol{Y}_{0:M} \mid \boldsymbol{\theta})$ is then computed by evaluating

$$\hat{p}(\boldsymbol{Y}_{0:M} \mid \boldsymbol{\theta}) = \frac{p(\boldsymbol{X}_{0:M}, \boldsymbol{Y}_{0:M} \mid \boldsymbol{\theta})}{\hat{p}(\boldsymbol{X}_{0:M} \mid \boldsymbol{Y}_{0:M}, \boldsymbol{\theta})}. \tag{4.14}$$

where the ratio may, in principle, be evaluated at any value of $\boldsymbol{X}_{0:M}$, though $\hat{\boldsymbol{X}}_{0:M}$ is used for numerical stability. We use the existing implementation of this method provided by the **TMB** package (Kristensen et al., 2016). Unlike other automatic differentiation (AD) libraries, **TMB** provides AD algorithms for (4.12) which efficiently leverage the sparsity of $\hat{\boldsymbol{V}}_{\boldsymbol{X}}$, which in turn allows for fast evaluations of the score and Hessian of the marginal likelihood in (4.14).

4.2.4 Max-and-Smooth

Another non-variational method is the Max-and-Smooth approach proposed by Hrafnkelsson et al. (2021), which is a powerful approximation to the marginal likelihood $p(\boldsymbol{\theta} \mid \boldsymbol{Y}_{0:M})$ for certain random-effects models. In this work, we generalize the approach to handle mixed-effects models as well. The mixed-effects model for SSMs is given by

$$\begin{aligned} \boldsymbol{X}_0^{(j)} &\sim \pi(\boldsymbol{X}_0^{(j)} \mid \boldsymbol{\eta}^{(j)}, \boldsymbol{\nu}) \\ \boldsymbol{X}_m^{(j)} &\sim p(\boldsymbol{X}_m^{(j)} \mid \boldsymbol{X}_{m-1}^{(j)}, \boldsymbol{\eta}^{(j)}, \boldsymbol{\nu}) \\ \boldsymbol{Y}_m^{(j)} &\overset{\text{iid}}{\sim} p(\boldsymbol{Y}_m^{(j)} \mid \boldsymbol{X}(t'_m)^{(j)}, \boldsymbol{\eta}^{(j)}, \boldsymbol{\nu}) \end{aligned} \tag{4.15}$$

with fixed effects $\boldsymbol{\nu}$ and random effects $\boldsymbol{\eta}^{(j)}$ following a multivariate normal distribution

$$\boldsymbol{\eta}^{j} \overset{\text{iid}}{\sim} \text{Normal}(\boldsymbol{\mu}_{\boldsymbol{\eta}}, \boldsymbol{\Sigma}_{\boldsymbol{\eta}}), \tag{4.16}$$

for $j = 1, \ldots D$ different datasets.

For problems in which the primary interest lies in estimating the fixed effect and hyperparameters $\boldsymbol{\theta} = (\boldsymbol{\nu}, \boldsymbol{\mu}_{\boldsymbol{\eta}}, \boldsymbol{\Sigma}_{\boldsymbol{\eta}})$, Max-and-Smooth provides a fast approximation to the marginal likelihood

$$\begin{aligned} p(\boldsymbol{Y}_{0:M}^{(1:D)} \mid \boldsymbol{\theta}) &= \int p(\boldsymbol{Y}_{0:M}^{(1:D)}, \boldsymbol{\eta}^{(1:D)} \mid \boldsymbol{\theta}) \, \mathrm{d}\boldsymbol{\eta}^{(1:D)} \\ &= \int \prod_{j=1}^{D} p(\boldsymbol{Y}_{0:M}^{(j)} \mid \boldsymbol{\xi}^{(j)}) p(\boldsymbol{\eta}^{(j)} \mid \boldsymbol{\mu}_{\boldsymbol{\eta}}, \boldsymbol{\Sigma}_{\boldsymbol{\eta}}) \, \mathrm{d}\boldsymbol{\eta}^{(1:D)}, \end{aligned} \tag{4.17}$$

where $\boldsymbol{\xi}^{(j)} = (\boldsymbol{\eta}^{(j)}, \boldsymbol{\nu})$. This is accomplished by first approximating the log-likelihood of each individual dataset with a quadratic function in $\boldsymbol{\xi}^{(j)}$:

$$p(\boldsymbol{Y}_{0:M}^{(j)} \mid \boldsymbol{\xi}^{(j)}) \approx c(\boldsymbol{Y}_{0:M}^{(j)}) \exp\left(-\frac{1}{2}(\boldsymbol{\xi}^{(j)} - \hat{\boldsymbol{\xi}}^{(j)})^\top \hat{\boldsymbol{V}}^{(j)} (\boldsymbol{\xi}^{(j)} - \hat{\boldsymbol{\xi}}^{(j)})\right), \tag{4.18}$$

where $c(\boldsymbol{Y}_{0:M}^{(j)})$ is a function of the observed data only. For SSMs, this quadratic approximation can be obtained by estimating the maximum likelihood and Fisher information of $p(\boldsymbol{Y}_{0:M}^{(j)} \mid \boldsymbol{\xi}^{(j)})$ for each individual dataset, using either a particle filter or the Laplace approximation described earlier.

With these quadratic approximations, we may write

$$\begin{aligned} p(\boldsymbol{Y}_{0:M}^{(1:D)} \mid \boldsymbol{\theta}) \approx c(\boldsymbol{Y}_{0:M}^{(1:D)}) \times \int \exp\Bigg(\sum_{j=1}^{D} & -\frac{1}{2}(\boldsymbol{\xi}^{(j)} - \hat{\boldsymbol{\xi}}^{(j)})^\top \hat{\boldsymbol{V}}^{(j)} (\boldsymbol{\xi}^{(j)} - \hat{\boldsymbol{\xi}}^{(j)}) \\ & -\frac{1}{2}(\boldsymbol{\eta}^{(j)} - \boldsymbol{\mu}_{\boldsymbol{\eta}})^\top \boldsymbol{\Sigma}_{\boldsymbol{\eta}}^{-1} (\boldsymbol{\eta}^{(j)} - \boldsymbol{\mu}_{\boldsymbol{\eta}}) - \frac{1}{2}\log\det(\boldsymbol{\Sigma}_{\boldsymbol{\eta}}) \Bigg) \, \mathrm{d}\boldsymbol{\eta}^{(1:D)} \\ = c(\boldsymbol{Y}_{0:M}^{(1:D)}) \int \exp\left(-\frac{1}{2} Q(\boldsymbol{\eta}^{(1:D)}; \boldsymbol{\theta}) - \frac{D}{2}\log\det(\boldsymbol{\Sigma}_{\boldsymbol{\eta}}) \right) \mathrm{d}\boldsymbol{\eta}^{(1:D)}, \end{aligned} \tag{4.19}$$

where $Q(\boldsymbol{\eta}^{(1:D)}; \boldsymbol{\theta})$ is a quadratic function in $\boldsymbol{\eta}^{(1:D)}$, whose parameters depend on $\boldsymbol{\theta}$, $\hat{\boldsymbol{\xi}}^{(1:D)}$ and $\hat{\boldsymbol{V}}^{(1:D)}$. Since Q is positive-definite, the integral in (4.19) can be evaluated in closed form using standard properties of the multivariate normal distribution. While the analytical expression for Q is technically involved, the required quantities can be readily obtained in practice using AD–for example, by applying a single step of the Newton–Raphson algorithm and computing the Hessian at the mode of the likelihood approximation.

4.2.5 Variational Inference

Bayesian inference involves estimating the joint posterior distribution using the SSM formulation,

$$\begin{aligned} p(\boldsymbol{\theta}, \boldsymbol{X}_{0:M} \mid \boldsymbol{Y}_{0:M}) &\propto p(\boldsymbol{\theta}, \boldsymbol{X}_{0:M}, \boldsymbol{Y}_{0:M}) \\ &= p(\boldsymbol{\theta}) \pi(\boldsymbol{X}_0 \mid \boldsymbol{\theta}) p(\boldsymbol{Y}_0 \mid \boldsymbol{X}_0, \boldsymbol{\theta}) \prod_{m=1}^{M} p(\boldsymbol{Y}_m \mid \boldsymbol{X}_m, \boldsymbol{\theta}) p(\boldsymbol{X}_m \mid \boldsymbol{X}_{m-1}, \boldsymbol{\theta}). \end{aligned} \tag{4.20}$$

Although this joint posterior can be sampled using MCMC methods, such approaches can be computationally expensive in high-dimensional settings, often requiring several days to obtain a sufficient number of samples (Whitaker et al., 2017).

A more scalable alternative is variational inference, which transforms the sampling problem into an optimization problem. Rather than drawing samples directly from $p(\boldsymbol{\theta}, \boldsymbol{X}_{0:M} \mid \boldsymbol{Y}_{0:M})$, we approximate it with a parameterized family of distributions $q(\boldsymbol{\theta}, \boldsymbol{X}_{0:M} \mid \boldsymbol{Y}_{0:M}; \boldsymbol{\phi})$ for variational parameters $\boldsymbol{\phi}$. The optimal approximation is found by minimizing the Kullback-Leibler (KL)-divergence $\mathrm{KL}\big(q(\boldsymbol{\theta}, \boldsymbol{X}_{0:M} \mid \boldsymbol{Y}_{0:M}; \boldsymbol{\phi}) \parallel p(\boldsymbol{\theta}, \boldsymbol{X}_{0:M} \mid \boldsymbol{Y}_{0:M})\big)$, which is equivalent to maximizing the ELBO (Jordan et al., 1999), defined as

$$\arg\max_{\boldsymbol{\phi}} \mathcal{L}(\boldsymbol{\phi}) := E_{q(\cdot;\boldsymbol{\phi})}[\log p(\boldsymbol{\theta}, \boldsymbol{X}_{0:M}, \boldsymbol{Y}_{0:M}) - \log q(\boldsymbol{\theta}, \boldsymbol{X}_{0:M} \mid \boldsymbol{Y}_{0:M}; \boldsymbol{\phi})]. \tag{4.21}$$

The joint log-density $\log p(\boldsymbol{\theta}, \boldsymbol{X}_{0:M}, \boldsymbol{Y}_{0:M})$ is approximated using the factorization in (4.20).

Most standard machine learning optimization algorithms require the gradient of the objective function $\mathcal{L}(\boldsymbol{\phi})$. However, the gradient of the expectation in (4.21) cannot be computed analytically. We, therefore, resort to Monte Carlo methods to approximate it. While it is possible to sample directly from $q(\boldsymbol{\theta}, \boldsymbol{X}_{0:M} \mid \boldsymbol{Y}_{0:M}; \boldsymbol{\phi})$, doing so prevents gradients from being propagated through the sampling operation. This, in turn, restricts the use of standard gradient-based optimization techniques such as stochastic gradient descent, Adam (Kingma and Ba, 2017), and their variants.

One approach to address this issue is the reparameterization trick (Kingma and Welling, 2022; Rezende et al., 2014; Titsias and Lázaro-Gredilla, 2014), which proceeds as follows. First, we sample $\boldsymbol{\epsilon}^{(1)}, \ldots, \boldsymbol{\epsilon}^{(S)} \sim p(\boldsymbol{\epsilon})$ from a distribution (e.g., standard Gaussian) that is independent of the variational parameters $\boldsymbol{\phi}$ and easy to sample from. The variational distribution $q(\boldsymbol{\theta}, \boldsymbol{X}_{0:M} \mid \boldsymbol{Y}_{0:M}; \boldsymbol{\phi})$ is then chosen such that each sample $(\boldsymbol{\theta}^{(i)}, \boldsymbol{X}_{0:M}^{(i)})$ can be expressed via a deterministic transformation,

$$(\boldsymbol{\theta}^{(i)}, \boldsymbol{X}_{0:M}^{(i)}) \sim q(\boldsymbol{\theta}, \boldsymbol{X}_{0:M} \mid \boldsymbol{Y}_{0:M}; \boldsymbol{\phi}) \quad \Longleftrightarrow \quad (\boldsymbol{\theta}^{(i)}, \boldsymbol{X}_{0:M}^{(i)}) = \boldsymbol{g}(\boldsymbol{\epsilon}^{(i)}, \boldsymbol{Y}_{0:M}, \boldsymbol{\phi}), \tag{4.22}$$

for $i = 1, \ldots, S$, where $\boldsymbol{g}$ is a function parameterized by $\boldsymbol{\phi}$ that defines this transformation.

The point is that all randomness is isolated in the pre-sampled noise variables $\boldsymbol{\epsilon}_1, \ldots, \boldsymbol{\epsilon}_S$ when sampling from $q(\boldsymbol{\theta}, \boldsymbol{X}_{0:M} \mid \boldsymbol{Y}_{0:M}; \boldsymbol{\phi})$ and the log-density becomes a deterministic and differentiable function of $\boldsymbol{\phi}$. This allows us to construct a straightforward and unbiased Monte Carlo estimator given by

$$\hat{\mathcal{L}}(\boldsymbol{\phi}) = \frac{1}{S} \sum_{i=1}^{S} [\log p(\boldsymbol{\theta}^{(i)}, \boldsymbol{X}_{0:M}^{(i)}, \boldsymbol{Y}_{0:M}) - \log q(\boldsymbol{\theta}^{(i)}, \boldsymbol{X}_{0:M}^{(i)} \mid \boldsymbol{Y}_{0:M}; \boldsymbol{\phi})]. \tag{4.23}$$

The gradient of $\hat{\mathcal{L}}(\boldsymbol{\phi})$ can be computed efficiently using backpropagation.

4.3 Information-Efficient Variational Inference

For ease of presentation, we begin by considering the case where the inter-observation step size equals the discretization step size, i.e., a resolution of $r = 1$. We approximate the conditional joint posterior distribution $p(\boldsymbol{\theta}, \boldsymbol{X}_{0:M} \mid \boldsymbol{Y}_{0:M})$ using the factorization given by

$$q(\boldsymbol{\theta}, \boldsymbol{X}_{0:M} \mid \boldsymbol{Y}_{0:M}; \boldsymbol{\phi}) = q(\boldsymbol{\theta} \mid \boldsymbol{Y}_{0:M}; \boldsymbol{\phi}_{\boldsymbol{\theta}}) q(\boldsymbol{X}_{0:M} \mid \boldsymbol{Y}_{0:M}, \boldsymbol{\theta}; \boldsymbol{\phi}_{\boldsymbol{X}}). \tag{4.24}$$

Here, the first term $q(\boldsymbol{\theta} \mid \boldsymbol{Y}_{0:M}; \boldsymbol{\phi}_{\boldsymbol{\theta}})$ serves as a variational approximation to the marginal posterior $p(\boldsymbol{\theta} \mid \boldsymbol{Y}_{0:M})$, while the second term $q(\boldsymbol{X}_{0:M} \mid \boldsymbol{Y}_{0:M}, \boldsymbol{\theta}; \boldsymbol{\phi}_{\boldsymbol{X}})$ approximates the conditional latent posterior $p(\boldsymbol{X}_{0:M} \mid \boldsymbol{Y}_{0:M}, \boldsymbol{\theta})$. Our goal is to optimize the variational parameter $\boldsymbol{\phi} = (\boldsymbol{\phi}_{\boldsymbol{\theta}}, \boldsymbol{\phi}_{\boldsymbol{X}})$ by maximizing the ELBO in (4.21) using the Monte Carlo estimator in (4.23). To do so efficiently, we choose distributions that are fast to sample from and compatible with the reparameterization trick in (4.22).

The choice of the variational family for $q(\boldsymbol{\theta} \mid \boldsymbol{Y}_{0:M}; \boldsymbol{\phi}_{\boldsymbol{\theta}})$ is flexible and often dependent on the SDE model. As such, we will select the most appropriate variational family in the experiments of Section 4.5.

4.3.1 Latent Variational Distribution

The latent variational distribution used in IEVI is derived from a linear Gaussian SSM for the latent dynamics with an arbitrary observation model:

$$\begin{aligned} \boldsymbol{X}_0 &\sim \text{Normal}(\boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0), \\ \boldsymbol{X}_m &\sim \text{Normal}(\boldsymbol{Q}_{m-1}\boldsymbol{X}_{m-1} + \boldsymbol{c}_{m-1}, \boldsymbol{R}_{m-1}), \\ \boldsymbol{Y}_m &\overset{\text{iid}}{\sim} p(\boldsymbol{Y}_m \mid \boldsymbol{X}_m, \boldsymbol{\theta}), \end{aligned} \tag{4.25}$$

where the parameters $\boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0, \boldsymbol{Q}_{0:M-1}, \boldsymbol{c}_{0:M-1}, \boldsymbol{R}_{0:M-1}$ are functions of $\boldsymbol{\theta}$ and $\boldsymbol{\phi}_{\boldsymbol{X}}$. Letting $q(\boldsymbol{X}_{0:M}, \boldsymbol{Y}_{0:M} \mid \boldsymbol{\theta}; \boldsymbol{\phi}_{\boldsymbol{X}})$ denote the joint distribution of the latent and observed variables in (4.25), the variational distribution we use, $q(\boldsymbol{X}_{0:M} \mid \boldsymbol{Y}_{0:M}, \boldsymbol{\theta}; \boldsymbol{\phi}_{\boldsymbol{X}})$, is the corresponding conditional distribution of the latents given the observations.

When the observation model $p(\boldsymbol{Y}_m \mid \boldsymbol{X}_m, \boldsymbol{\theta})$ is also linear and Gaussian, this conditional distribution can be computed efficiently using Kalman filtering and smoothing recursions. In the general case, however, the filtering distribution becomes intractable and must be approximated using a Gaussian distribution with mean $\boldsymbol{\mu}_{m|m}$ and variance $\boldsymbol{\Sigma}_{m|m}$ in order to apply Kalman smoothing (Rauch et al., 1965; Särkkä, 2013). A key challenge with this

formulation is that the number of parameters, $\boldsymbol{Q}_m, \boldsymbol{c}_m, \boldsymbol{R}_m, \boldsymbol{\mu}_{m|m}, \boldsymbol{\Sigma}_{m|m}$, scales with the number of time points. To address this, we introduce a RNN architecture to parameterize these quantities more compactly.

More precisely, we use a RNN to estimate the filtering and transition quantities $\boldsymbol{\mu}_{m|m}$, $\boldsymbol{\Sigma}_{m|m}, \boldsymbol{Q}_m, \boldsymbol{c}_m$ and $\boldsymbol{R}_m$ at each time step $m = 0, \ldots M$. Formally, in its unrolled representation, the RNN updates its hidden state and outputs as follows:

$$\boldsymbol{o}_m, \boldsymbol{h}_{m+1} = \mathrm{RNN}(\boldsymbol{Y}_m, \boldsymbol{\theta}, \boldsymbol{h}_m; \boldsymbol{\phi}_{\boldsymbol{X}}), \tag{4.26}$$

where the output $\boldsymbol{o}_m$ is a vector containing $\big(\boldsymbol{\mu}_{m|m}, \boldsymbol{\Sigma}_{m|m}, \boldsymbol{Q}_m, \boldsymbol{c}_m, \boldsymbol{R}_m\big)$. Each hidden state $\boldsymbol{h}_m$ summarizes the observation history up to time $m-1$, having been computed recursively from the previous hidden state $\boldsymbol{h}_{m-1}$ and the observation $\boldsymbol{Y}_{m-1}$. Combined with the current observation $\boldsymbol{Y}_m$, the RNN ensures that $\boldsymbol{h}_{m+1}$ and $\boldsymbol{o}_m$ condition on all observations up to time m, as required by the filtering distribution. Figure 4.1 illustrates the RNN's unrolled structure, showing the hidden state transitions and corresponding outputs at each time step.

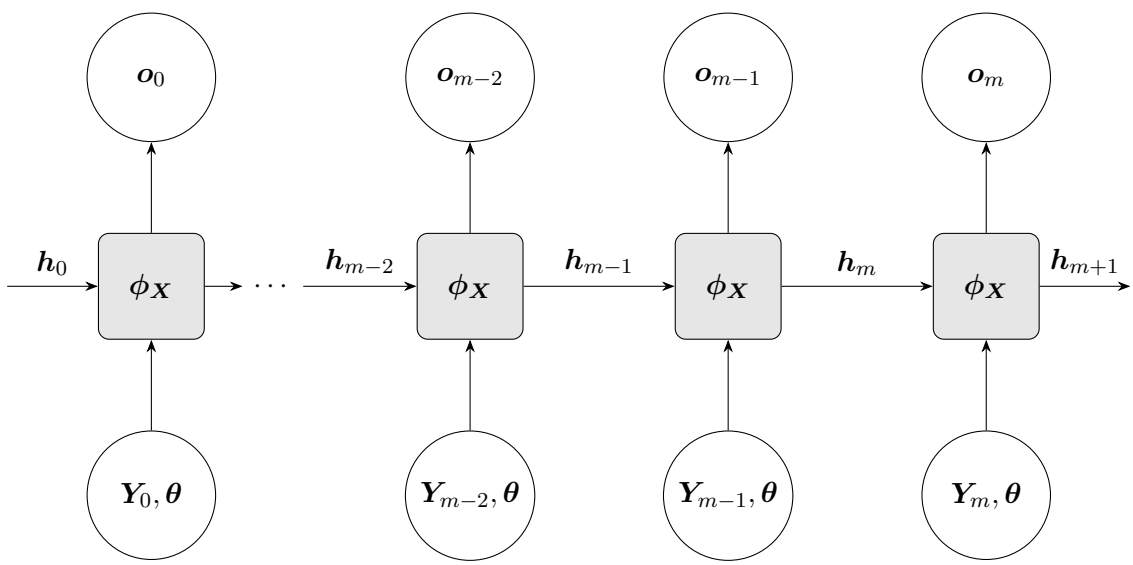

Figure 4.1: Unrolled RNN from time step 0 to m, where each cell updates the hidden state using the current observation and produces corresponding outputs.

Given the filtering distribution, the corresponding smoothing distribution $q(\boldsymbol{X}_m \mid \boldsymbol{Y}_{0:M}, \boldsymbol{\theta}; \boldsymbol{\phi}_{\boldsymbol{X}})$ can be efficiently sampled using the Kalman backward pass, based on the decomposi-

tion (Doucet and Johansen, 2009; Särkkä, 2013, Chapter 8),

$$
\begin{aligned}
q(\boldsymbol{X}_{0:M} \mid \boldsymbol{Y}_{0:M}, \boldsymbol{\theta}; \boldsymbol{\phi}_{\boldsymbol{X}}) &= q(\boldsymbol{X}_M \mid \boldsymbol{Y}_{0:M}, \boldsymbol{\theta}; \boldsymbol{\phi}_{\boldsymbol{X}}) \prod_{m=0}^{M-1} q(\boldsymbol{X}_m \mid \boldsymbol{X}_{m+1}, \boldsymbol{Y}_{0:M}, \boldsymbol{\theta}; \boldsymbol{\phi}_{\boldsymbol{X}}) \\
&= q(\boldsymbol{X}_M \mid \boldsymbol{Y}_{0:M}, \boldsymbol{\theta}; \boldsymbol{\phi}_{\boldsymbol{X}}) \prod_{m=0}^{M-1} q(\boldsymbol{X}_m \mid \boldsymbol{X}_{m+1}, \boldsymbol{Y}_{0:m}, \boldsymbol{\theta}; \boldsymbol{\phi}_{\boldsymbol{X}}).
\end{aligned} \tag{4.27}
$$

Sampling from the variational smoothing distribution proceeds recursively using the Kalman backward pass. At time M, we draw

$$
\boldsymbol{X}_M \sim q(\boldsymbol{X}_M \mid \boldsymbol{Y}_{0:M}, \boldsymbol{\theta}; \boldsymbol{\phi}_{\boldsymbol{X}}) = \text{Normal}(\boldsymbol{\mu}_{M|M}, \boldsymbol{\Sigma}_{M|M}), \tag{4.28}
$$

directly from the variational filtering distribution. Here, we reuse the notation $\boldsymbol{\mu}_{n|m} = E[\boldsymbol{X}(n\Delta t) \mid \boldsymbol{Y}_{0:m}]$ and $\boldsymbol{\Sigma}_{n|m} = \text{var}[\boldsymbol{X}(n\Delta t) \mid \boldsymbol{Y}_{0:m}]$ introduced in Chapter 2. Then, for each time step $m = M-1, \ldots 0$, assuming that $\boldsymbol{X}_{m+1}$ has already been sampled, the conditional distribution $q(\boldsymbol{X}_m \mid \boldsymbol{X}_{m+1}, \boldsymbol{Y}_{0:M}, \boldsymbol{\theta}; \boldsymbol{\phi}_{\boldsymbol{X}})$ is also Gaussian, with mean and variance given by (Rauch et al., 1965; Särkkä, 2013)

$$
\begin{aligned}
\boldsymbol{A}_m &= \boldsymbol{\Sigma}_{m|m} \boldsymbol{Q}_{m+1}^\top \boldsymbol{\Sigma}_{m+1|m}^{-1} \\
\tilde{\boldsymbol{\mu}}_{m|M} &= \boldsymbol{\mu}_{m|m} + \boldsymbol{A}_m (\boldsymbol{X}_{m+1} - \boldsymbol{\mu}_{m+1|m}) \\
\tilde{\boldsymbol{\Sigma}}_{m|M} &= \boldsymbol{\Sigma}_{m|m} - \boldsymbol{A}_m \boldsymbol{Q}_{m+1} \boldsymbol{\Sigma}_{m|m}.
\end{aligned} \tag{4.29}
$$

The combination of the RNN-based filtering pass and the Kalman backward smoothing pass defines a deterministic transformation of the observations $\boldsymbol{Y}_{0:M}$, SDE parameters $\boldsymbol{\theta}$, variational parameters $\boldsymbol{\phi}_{\boldsymbol{X}}$, and Gaussian noise $\boldsymbol{\epsilon}_{\boldsymbol{X}}$. Specifically, given $\boldsymbol{\phi}_{\boldsymbol{X}}$, $\boldsymbol{\theta}$ and $\boldsymbol{Y}_{0:M}$, the RNN produces a sequence of estimates $(\boldsymbol{\mu}_{m|m}, \boldsymbol{\Sigma}_{m|m}, \boldsymbol{Q}_m, \boldsymbol{c}_m, \boldsymbol{R}_m)$, which are then used in the backward pass to compute the smoothed means $\tilde{\boldsymbol{\mu}}_{m|M}$ and variances $\tilde{\boldsymbol{\Sigma}}_{m|M}$ in (4.29). Since each conditional distribution is Gaussian, the full sampling process for $\boldsymbol{X}_{0:M}$ can be expressed as

$$
\begin{aligned}
\boldsymbol{X}_m^{(i)} &= \boldsymbol{g}_{\boldsymbol{X}}(\boldsymbol{\epsilon}_{\boldsymbol{X}}^{(i,m)}, \boldsymbol{X}_{m+1}^{(i)}, \boldsymbol{\theta}, \boldsymbol{Y}_{0:M}, \boldsymbol{\phi}_{\boldsymbol{X}}) \\
&= \tilde{\boldsymbol{\mu}}_{m|M} + \tilde{\boldsymbol{\Sigma}}_{m|M}^{1/2} \boldsymbol{\epsilon}_{\boldsymbol{X}}^{(i,m)},
\end{aligned} \tag{4.30}
$$

where $\boldsymbol{\epsilon}_{\boldsymbol{X}}^{(i,m)} \overset{\text{ind}}{\sim} \text{Normal}(\boldsymbol{0}, \boldsymbol{I}_d)$ for $m = 0, \ldots, M$ and $i = 1, \ldots, S$, and d is the dimension of the latent state $\boldsymbol{X}_m$. Because all distributions involved are easy-to-sample-from Gaussians, and the reparameterization trick is applied via (4.30), the RNN can be efficiently trained by maximizing the ELBO in (4.23) with stochastic gradient descent.

4.3.2 Linear Gaussian Observation Model

As a special case, we consider a linear Gaussian observation model, following Golightly and Wilkinson (2008); Ryder et al. (2018), given by

$$\boldsymbol{Y}_m \overset{\text{iid}}{\sim} \text{Normal}(\boldsymbol{W}\boldsymbol{X}_m, \boldsymbol{\Sigma}_{\boldsymbol{\theta}}), \tag{4.31}$$

where $\boldsymbol{W}$ is a fixed matrix, and $\boldsymbol{\Sigma}_{\boldsymbol{\theta}}$ is the observation noise, which may be either known or treated as an unknown parameter.

As noted earlier, when the observation model is linear and Gaussian, the variational filtering distribution $q(\boldsymbol{X}_m \mid \boldsymbol{Y}_{0:m}, \boldsymbol{\theta}; \boldsymbol{\phi}_{\boldsymbol{X}})$ is Gaussian and can be computed exactly using the Kalman filter. In this case, the RNN in (4.26) no longer needs to estimate the filtering moments $\boldsymbol{\mu}_{m|m}$ and $\boldsymbol{\Sigma}_{m|m}$, since these quantities are available in closed form.

The simplified RNN output is given by

$$\boldsymbol{o}_m, \boldsymbol{h}_{m+1} = \text{RNN}(\boldsymbol{Y}_m, \boldsymbol{\theta}, \boldsymbol{h}_m; \boldsymbol{\phi}_{\boldsymbol{X}}), \tag{4.32}$$

where $\boldsymbol{o}_m$ is a vector containing only the transition parameters $(\boldsymbol{Q}_m, \boldsymbol{c}_m, \boldsymbol{R}_m)$. The rest of the procedure for sampling from the variational smoothing distribution $q(\boldsymbol{X}_m \mid \boldsymbol{Y}_{0:M}, \boldsymbol{\theta}; \boldsymbol{\phi}_{\boldsymbol{X}})$ proceeds exactly as described in Section 4.3.1.

4.3.3 Higher Resolution

Earlier, we assumed that the discretization step size Δt is equal to the inter-observation step size $\Delta t'$. However, in practice, a sufficiently accurate approximation of the latent dynamics often requires using a smaller step size. As an example, consider the case where the resolution is $r = 3$, as illustrated in Figure 4.2. In this setting, the latent variables $\boldsymbol{X}(0)$ and $\boldsymbol{X}(3\Delta t)$ correspond to the times t'_0 and t'_1, at which the measurements $\boldsymbol{Y}_0$ and $\boldsymbol{Y}_1$ are recorded, respectively.

When $r = 1$, there is one observation per latent time step, and we can directly use the observation $\boldsymbol{Y}_m$ and SDE model parameter $\boldsymbol{\theta}$ as inputs to train the RNN. However, when $r > 1$, the observations are more sparsely distributed relative to the latent discretization. For each discretized latent time point $t_n = n\Delta t$, let $t'_{-(n)}$ denote the most recent observation time at or before t_n, and $t'_{+(n)}$ the next observation time after t_n. In our experiments, the following inputs worked well for approximating the latent process at time t_n:

- The parameter: $\boldsymbol{\theta}$

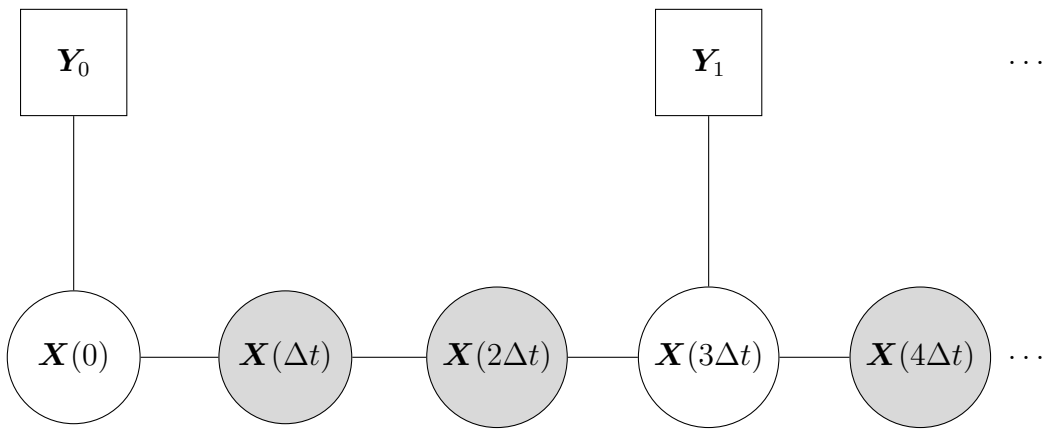

Figure 4.2: Illustration of resolution $r = 3$, where $\Delta t = \Delta t'/3$.

- The most recent observation: $\boldsymbol{Y}_{-(n)} := \boldsymbol{Y}(t'_{-(n)})$
- The next available observation: $\boldsymbol{Y}_{+(n)} := \boldsymbol{Y}(t'_{+(n)})$
- The time until the next observation: $t'_{+(n)} - t_n$

For example, in Figure 4.2, the RNN uses $\boldsymbol{Y}_0$ and $\boldsymbol{Y}_1$ as inputs to approximate the latent variables $\boldsymbol{X}(0), \boldsymbol{X}(\Delta t)$ and $\boldsymbol{X}(2\Delta t)$. Including the next observation as an input provides the RNN with information about the direction of the latent process, thereby improving the accuracy of the inferred latent states. Algorithm 4.2 outlines the full procedure for sampling from $q(\boldsymbol{X}_{0:M} \mid \boldsymbol{Y}_{0:M}, \boldsymbol{\theta}; \boldsymbol{\phi})$ and computing its log-density under a general observation model. Algorithm 4.3 describes the corresponding procedure assuming a linear Gaussian observation model.

4.4 Related Works

Archer et al. (2015) propose a black-box variational inference method for SSMs, where the posterior $p(\boldsymbol{X}_{0:M} \mid \boldsymbol{Y}_{0:M}, \hat{\boldsymbol{\theta}})$ is approximated by a variational distribution $q(\boldsymbol{X}_{0:M}; \hat{\boldsymbol{\phi}})$. Here, $\hat{\boldsymbol{\theta}}$ is the mode of the likelihood $p(\boldsymbol{Y}_{0:M} \mid \boldsymbol{\theta})$, and $\hat{\boldsymbol{\phi}}$ is the variational parameter that maximizes the ELBO. Their variational family is a multivariate normal distribution with a block tridiagonal inverse variance matrix, enabling sampling with linear time complexity in the number of observations. However, their method conditions each latent state on only nearby observations, rather than on all observations $\boldsymbol{Y}_{0:M}$, as required for smoothing.

Algorithm 4.2 The IEVI method for drawing a single sample $\boldsymbol{X}_{0:M} \sim q(\boldsymbol{X}_{0:M} \mid \boldsymbol{Y}_{0:M}, \boldsymbol{\theta}; \boldsymbol{\phi})$ and computing its log-density assuming an arbitrary observation model.

procedure `ievi`$(\boldsymbol{Y}_{0:M}, \boldsymbol{\theta})$
  $\boldsymbol{h}_0 \leftarrow \boldsymbol{0}$ ▷ Initialization for hidden state at 0.
  $\ell_{\boldsymbol{X}} \leftarrow 0$

  for $n = 0 : N$ **do**
    $(\boldsymbol{\mu}_{n|n}, \boldsymbol{\Sigma}_{n|n}, \boldsymbol{Q}_n, \boldsymbol{c}_n, \boldsymbol{R}_n), \boldsymbol{h}_{n+1} \leftarrow \mathrm{RNN}(\boldsymbol{Y}_{-(n)}, \boldsymbol{Y}_{+(n)}, t'_{+(n)} - t_n, \boldsymbol{\theta}, \boldsymbol{h}_n; \boldsymbol{\phi})$
    $\boldsymbol{\mu}_{n+1|n}, \boldsymbol{\Sigma}_{n+1|n} \leftarrow$ `kalman_predict`$(\boldsymbol{\mu}_{n|n}, \boldsymbol{\Sigma}_{n|n}, \boldsymbol{c}_n, \boldsymbol{Q}_n, \boldsymbol{R}_n)$

  $\boldsymbol{X}(N\Delta t) \sim \mathrm{Normal}(\boldsymbol{\mu}_{N|N}, \boldsymbol{\Sigma}_{N|N})$
  $\ell_{\boldsymbol{X}} \leftarrow$ `normal_logpdf`$(\boldsymbol{X}(N\Delta t); \boldsymbol{\mu}_N, \boldsymbol{\Sigma}_N)$
  for $n = N - 1 : 0$ **do**
    $\boldsymbol{\mu}_n, \boldsymbol{\Sigma}_n \leftarrow$ `kalman_sample`$(\boldsymbol{X}((n+1)\Delta t), \boldsymbol{\mu}_{n|n}, \boldsymbol{\Sigma}_{n|n}, \boldsymbol{\mu}_{n+1|n}, \boldsymbol{\Sigma}_{n+1|n}, \boldsymbol{Q}_{n+1})$
    $\boldsymbol{X}(n\Delta t) \sim \mathrm{Normal}(\boldsymbol{\mu}_n, \boldsymbol{\Sigma}_n)$
    $\ell_{\boldsymbol{X}} \leftarrow \ell_{\boldsymbol{X}} +$ `normal_logpdf`$(\boldsymbol{X}(n\Delta t); \boldsymbol{\mu}_n, \boldsymbol{\Sigma}_n)$

  return $\boldsymbol{X}_{0:M}, \ell_{\boldsymbol{X}}$

Another approach, proposed by Ryder et al. (2018), samples from the variational distribution using the factorization

$$q(\boldsymbol{\theta}, \boldsymbol{X}_{0:M}; \boldsymbol{\phi}) = q(\boldsymbol{\theta}; \boldsymbol{\phi}) q(\boldsymbol{X}_{0:M} \mid \boldsymbol{\theta}; \boldsymbol{\phi}). \tag{4.33}$$

They assume a mean-field Gaussian for $q(\boldsymbol{\theta}; \boldsymbol{\phi})$. To model $q(\boldsymbol{X}_{0:M} \mid \boldsymbol{\theta}; \boldsymbol{\phi})$, they mimic the Euler-Maruyama scheme by using a RNN to approximate the drift $\hat{\boldsymbol{\alpha}}$ and the diffusion $\hat{\boldsymbol{\beta}}$. At each time step n, the RNN uses the parameter $\boldsymbol{\theta}$, and the observations up to time n as input, but does not have access to future observations.

Our work addresses two key limitations of this approach. First, we construct a variational smoothing distribution over the latent variables that conditions on both past and future observations while maintaining linear time complexity. Second, whereas the method proposed by Ryder et al. (2018) requires processing the entire observation sequence end-to-end, our framework allows for flexible batching over both the number of discretization steps and the number of individual datasets in mixed-effects models. This enables training to scale efficiently along both dimensions. As batching is known to accelerate convergence in practice (Goodfellow et al., 2016, Chapter 8), our method remains computationally efficient even for large datasets.

Algorithm 4.3 The IEVI method for drawing a single sample $\boldsymbol{X}_{0:M} \sim q(\boldsymbol{X}_{0:M} \mid \boldsymbol{Y}_{0:M}, \boldsymbol{\theta}; \boldsymbol{\phi})$ and computing its log-density assuming linear Gaussian observation model. $\boldsymbol{W}$ is a fixed matrix and $\boldsymbol{\mu}_0$ and $\boldsymbol{\Sigma}_0$ are the prior parameters for $\pi(\boldsymbol{X}_0)$ in (4.25), which may be estimated as a part of $\boldsymbol{\theta}$.

procedure `ievi_gauss`$(\boldsymbol{Y}_{0:M}, \boldsymbol{\theta}, \boldsymbol{W})$
  $\boldsymbol{h}_0 \leftarrow \boldsymbol{0}$ ▷ Initialization for hidden state at 0.
  $\ell_{\boldsymbol{X}} \leftarrow 0$
  ▷ Update with observation $\boldsymbol{Y}_0$.
  $\boldsymbol{\mu}_{0|0}, \boldsymbol{\Sigma}_{0|0} \leftarrow$ `kalman_update`$(\boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0, \boldsymbol{Y}_0, \boldsymbol{0}, \boldsymbol{W}, \boldsymbol{\Sigma}_{\boldsymbol{\theta}})$

  for $n = 0 : N-1$ **do**
    $(\boldsymbol{Q}_n, \boldsymbol{c}_n, \boldsymbol{R}_n), \boldsymbol{h}_{n+1} \leftarrow \mathrm{RNN}(\boldsymbol{Y}_{-(n)}, \boldsymbol{Y}_{+(n)}, t'_{+(n)} - t_n, \boldsymbol{\theta}, \boldsymbol{h}_n; \boldsymbol{\phi})$
    $\boldsymbol{\mu}_{n+1|n}, \boldsymbol{\Sigma}_{n+1|n} \leftarrow$ `kalman_predict`$(\boldsymbol{\mu}_{n|n}, \boldsymbol{\Sigma}_{n|n}, \boldsymbol{c}_n, \boldsymbol{Q}_n, \boldsymbol{R}_n)$
    ▷ Observation available at this time point.
    if $(n+1) \equiv 0 \mod r$ **then**
      $m \leftarrow (n+1)/r$ ▷ Index for the observation.
      $\boldsymbol{\mu}_{n+1|n+1}, \boldsymbol{\Sigma}_{n+1|n+1} \leftarrow$ `kalman_update`$(\boldsymbol{\mu}_{n+1|n}, \boldsymbol{\Sigma}_{n+1|n}, \boldsymbol{Y}_m, \boldsymbol{0}, \boldsymbol{W}, \boldsymbol{\Sigma}_{\boldsymbol{\theta}})$
    else
      $\boldsymbol{\mu}_{n+1|n+1}, \boldsymbol{\Sigma}_{n+1|n+1} \leftarrow \boldsymbol{\mu}_{n+1|n}, \boldsymbol{\Sigma}_{n+1|n}$ ▷ No update required.

  $\boldsymbol{X}(N\Delta t) \sim \mathrm{Normal}(\boldsymbol{\mu}_{N|N}, \boldsymbol{\Sigma}_{N|N})$
  $\ell_{\boldsymbol{X}} \leftarrow$ `normal_logpdf`$(\boldsymbol{X}(N\Delta t); \boldsymbol{\mu}_N, \boldsymbol{\Sigma}_N)$
  for $n = N-1 : 0$ **do**
    $\boldsymbol{\mu}_n, \boldsymbol{\Sigma}_n \leftarrow$ `kalman_sample`$(\boldsymbol{X}((n+1)\Delta t), \boldsymbol{\mu}_{n|n}, \boldsymbol{\Sigma}_{n|n}, \boldsymbol{\mu}_{n+1|n}, \boldsymbol{\Sigma}_{n+1|n}, \boldsymbol{Q}_{n+1})$
    $\boldsymbol{X}(n\Delta t) \sim \mathrm{Normal}(\boldsymbol{\mu}_n, \boldsymbol{\Sigma}_n)$
    $\ell_{\boldsymbol{X}} \leftarrow \ell_{\boldsymbol{X}} +$ `normal_logpdf`$(\boldsymbol{X}(n\Delta t); \boldsymbol{\mu}_n, \boldsymbol{\Sigma}_n)$

  return $\boldsymbol{X}_{0:M}, \ell_{\boldsymbol{X}}$

4.5 Numerical Experiments

In this section, we compare the performance of IEVI with the variational inference method proposed by Ryder et al. (2018), as well as two traditional non-variational approaches. All methods are evaluated from a Bayesian perspective in the context of parameter estimation across multiple experiments. Our primary goal is to approximate the posterior distribution $p(\boldsymbol{\theta} \mid \boldsymbol{Y}_{0:M})$.

IEVI uses the variational approximation defined in (4.24), while the method of Ryder et al. (2018) uses a similar factorization, as shown in (4.33). Both methods are implemented in **JAX** (Bradbury et al., 2018), with neural networks built using the **Equinox** library (Kidger and Garcia, 2021). The neural network weights used in $q(\boldsymbol{X}_{0:M} \mid \boldsymbol{\theta}, \boldsymbol{Y}_{0:M}; \boldsymbol{\phi}_{\boldsymbol{\theta}})$ are initialized randomly using Kaiming initialization (He et al., 2015), which is the default in **Equinox**. Variational parameters are optimized by maximizing the Monte Carlo estimator of the ELBO in (4.23), using the Adam optimizer (Kingma and Ba, 2017) as implemented in the **Optax** library (DeepMind et al., 2020).

For the non-variational methods that approximate the likelihood function $\mathcal{L}(\boldsymbol{\theta} \mid \boldsymbol{Y}_{0:M})$ described in Section 4.2, we approximate the parameter posterior distribution via

$$\hat{p}(\boldsymbol{\theta} \mid \boldsymbol{Y}_{0:M}) \propto p(\boldsymbol{\theta}) \times \hat{\mathcal{L}}(\boldsymbol{\theta} \mid \boldsymbol{Y}_{0:M}). \tag{4.34}$$

To sample from this approximate posterior $\hat{p}(\boldsymbol{\theta} \mid \boldsymbol{Y}_{0:M})$, we again use the Bayesian normal approximation introduced in Section 2.6.1 of Chapter 2, as it is orders of magnitude faster than alternatives such as MCMC. Specifically, we approximate the posterior as a multivariate normal,

$$\boldsymbol{\theta} \mid \boldsymbol{Y}_{0:M} \approx \text{Normal}(\hat{\boldsymbol{\theta}}, \hat{\boldsymbol{V}}), \tag{4.35}$$

where

$$\hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}} \log \hat{p}(\boldsymbol{\theta} \mid \boldsymbol{Y}_{0:M}), \quad \hat{\boldsymbol{V}} = -\left[\frac{\partial^2}{\partial \boldsymbol{\theta} \partial \boldsymbol{\theta}'} \log \hat{p}(\hat{\boldsymbol{\theta}} \mid \boldsymbol{Y}_{0:M})\right]^{-1}. \tag{4.36}$$

The posterior mode $\hat{\boldsymbol{\theta}}$ of the Bayesian normal approximation is estimated using the Broyden–Fletcher–Goldfarb–Shanno (BFGS) optimization algorithm (Nocedal and Wright, 2006), as implemented in the **JAXopt** library (Blondel et al., 2021). To estimate the posterior variance of $\boldsymbol{\theta}$, we compute the Hessian of $\log \hat{p}(\boldsymbol{\theta} \mid \boldsymbol{Y}_{0:M})$ at $\hat{\boldsymbol{\theta}}$ and take its inverse.

For each of the following experiments, we simulate data by sampling a trajectory from the SDE, approximated using the Euler-Maruyama scheme with a very small step size and the true parameter $\boldsymbol{\theta}$. Observation noise is then added to this trajectory using the default pseudo-random number generator in Python.

4.5.1 Lotka–Volterra Model

The Lotka-Volterra models describe the interaction between two species: one as the predator and the other as the prey (Lotka., 1925). The models assume that the prey population grows through reproduction and declines due to natural causes or predation by predators. Predators, on the other hand, rely on consuming prey to survive and reproduce and die

naturally. The dynamics of the system are governed by a set of differential equations that describe how changes in one population affect the other.

A SDE version of the model, proposed by Golightly and Wilkinson (2011), is given by

$$\mathrm{d}\boldsymbol{X}_t = \begin{bmatrix} \alpha U_t - \beta U_t V_t \\ \beta U_t V_t - \gamma \end{bmatrix} \mathrm{d}t + \begin{bmatrix} \alpha U_t + \beta U_t V_t & -\beta U_t V_t \\ -\beta U_t V_t & \gamma V_t + \beta U_t V_t \end{bmatrix}^{1/2} \mathrm{d}\boldsymbol{B}_t, \tag{4.37}$$

where $\boldsymbol{X}_t = (U_t, V_t)^\top$ denotes the prey and predator populations at time t, and $\boldsymbol{B}_t$ is a 2-dimensional standard Brownian motion. The parameters of interest are $\boldsymbol{\theta} = (\log \alpha, \log \beta, \log \gamma)$ on the unconstrained scale, for which we assume a flat prior $\pi(\boldsymbol{\theta}) \propto 1$. The observation model used by Ryder et al. (2018) assumes independent measurements according to

$$\boldsymbol{Y}_i \overset{\text{ind}}{\sim} \text{Normal}(\boldsymbol{X}(t_i), \boldsymbol{I}_{2x2}) \tag{4.38}$$

for $t_i = \{0, 10, 20, 30, 40\}$. We use the same setting for this problem as Ryder et al. (2018) for a direct comparison. In particular, we use a discretization resolution of $r = 100$, corresponding to a step size $\Delta t = 0.1$.

We first evaluate the quality of the approximation to $p(\boldsymbol{X}_{0:M} \mid \boldsymbol{Y}_{0:M}, \boldsymbol{\theta})$ under the assumption that $\boldsymbol{\theta}$ is known, using three approaches: our IEVI method, the method of Ryder et al. (2018) and particle filtering. For the variational methods, we use the variational distribution $q(\boldsymbol{X}_{0:M} \mid \boldsymbol{Y}_{0:M}, \boldsymbol{\theta}; \boldsymbol{\phi})$ as the approximation. All approximations are compared against the true latent trajectory, simulated using $\boldsymbol{\theta} = (\log 0.5, \log 0.0025, \log 0.3)$ and initial value $\boldsymbol{X}_0 = (71, 79)^\top$.

Figure 4.3 displays $S = 1000$ sample drawn from the variational distributions produced by IEVI and the method of Ryder et al. (2018). For particle filtering, we use $P = 100,000$ particles with a multinomial resampler, combined with the standard particle smoother (Kitagawa, 1996; Doucet and Johansen, 2009), which produces an accurate estimate of the true posterior $p(\boldsymbol{X}_{0:M} \mid \boldsymbol{Y}_{0:M}, \boldsymbol{\theta})$. Also shown are the ground truth latent trajectory and the observed data. As expected, all methods cover the latent states at the observation times. However, only IEVI provides good coverage of the latent trajectory between observation times. This suggests that, by conditioning on all observations, IEVI improves the posterior approximation compared to methods based solely on the filtering distribution.

We now turn to the parameter inference problem, where we use the variational distribution $q(\boldsymbol{\theta} \mid \boldsymbol{Y}_{0:M}; \boldsymbol{\phi}_{\boldsymbol{\theta}})$ to approximate the parameter posterior, and $q(\boldsymbol{X}_{0:M} \mid \boldsymbol{Y}_{0:M}, \boldsymbol{\theta}; \boldsymbol{\phi}_{\boldsymbol{X}})$ to approximate the latent distribution. For the parameter variational distribution, we use

$$q(\boldsymbol{\theta} \mid \boldsymbol{Y}_{0:M}; \boldsymbol{\phi}_{\boldsymbol{\theta}}) = \text{Normal}(\boldsymbol{\mu}_{\boldsymbol{\phi}}, \boldsymbol{\Sigma}_{\boldsymbol{\phi}}), \tag{4.39}$$

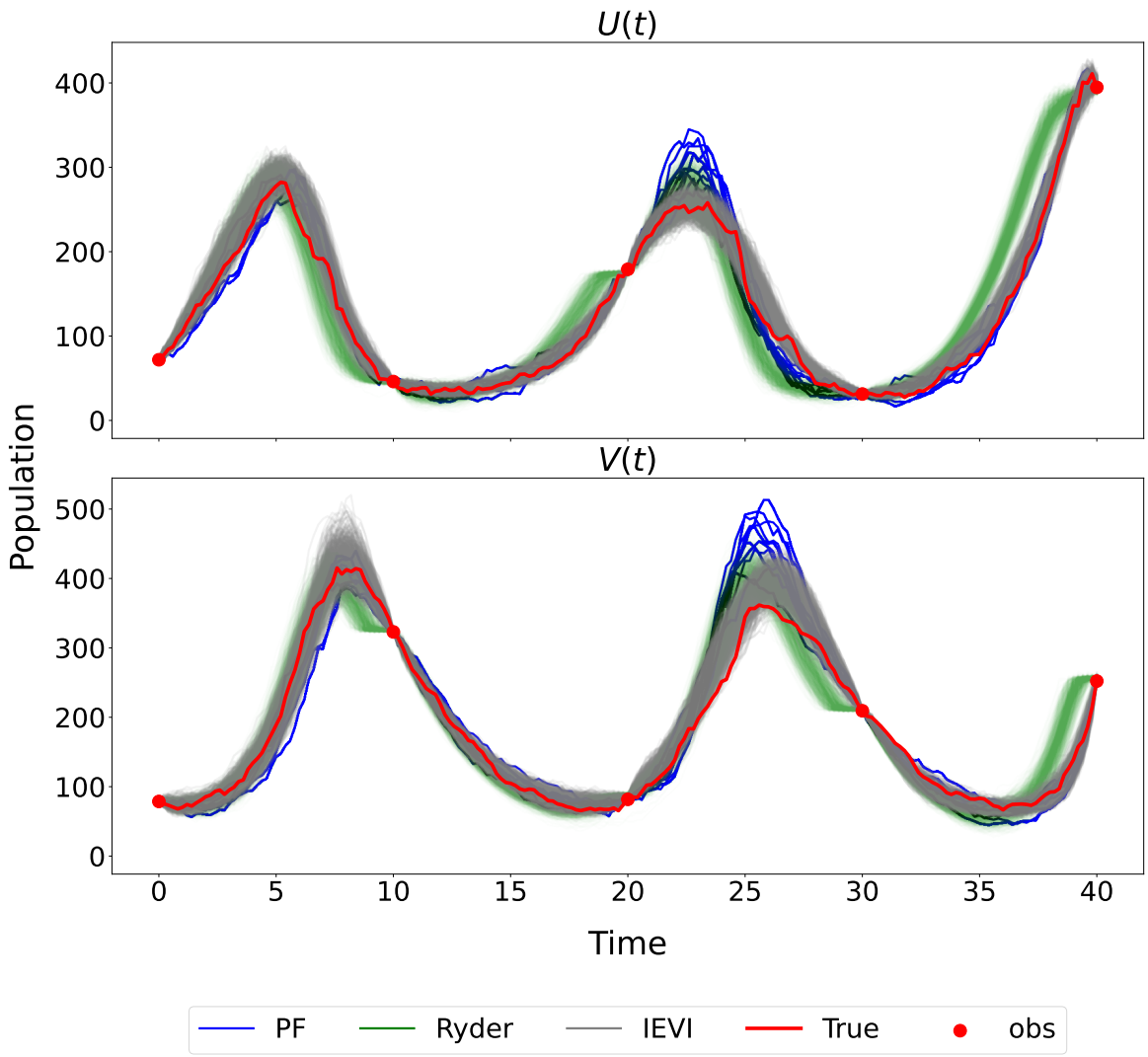

Figure 4.3: $S = 1000$ sample draws from the variational distributions of IEVI and the method of Ryder et al. (2018), and $P = 100,000$ particles from particle filtering, shown alongside the ground truth latent trajectory and observed data.

where $\boldsymbol{\phi}_{\boldsymbol{\theta}} = (\boldsymbol{\mu}_{\phi}, \boldsymbol{\Sigma}_{\phi}^{1/2})$ and $\boldsymbol{\Sigma}_{\phi}^{1/2}$ denote the Cholesky decomposition of the variance matrix $\boldsymbol{\Sigma}_{\phi}$. We operate on the Cholesky scale because a general variance matrix is symmetric and thus overparameterized. To ensure that the diagonal entries of the lower triangular matrix remain positive, we apply the softplus function, defined as $\text{softplus}(z) = \log(1 + e^{z})$. We

choose the softplus function over alternatives such as the exponentiation due to its superior numerical stability in our experiments.

Figure 4.4 presents the parameter posterior distributions estimated using the variational distributions of IEVI and the method of Ryder et al. (2018). The two non-variational methods estimate their posteriors via the Bayesian normal approximation. Among the four methods, only IEVI and the Laplace approximation produce posterior distributions that cover all three true parameter values, with the mode of the posterior from IEVI being closer to the true value of α. In contrast, the posteriors from particle filtering and the method of Ryder et al. (2018) fail to cover the true value of γ.

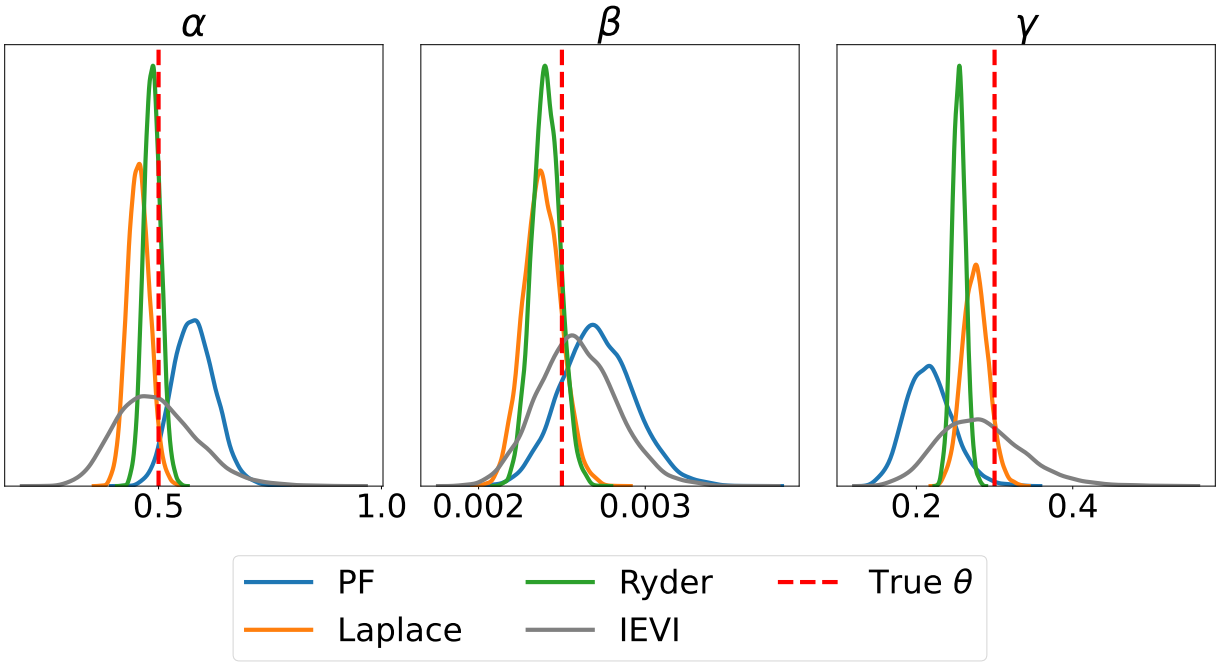

Figure 4.4: Parameter posteriors for the Lotka-Volterra model using various methods.

We further investigate the accuracy and reliability of each method across 100 different datasets. Table 4.1 displays the RMSE of the mode of each method's posterior relative to the true parameter values. The number in brackets indicates how many times the true parameter value was contained within the 95% credible interval across the 100 simulations. IEVI achieves the lowest RMSE for two out of the three parameters and demonstrates the highest coverage among all methods.

Table 4.1: The RMSE for each parameter of the Lotka-Volterra model across 100 simulations, using various inference methods. The number in brackets indicates how many times the true parameter value was contained within the 95% credible interval.

Method	α	β	γ
Particle Filtering	0.246 (29)	0.0015 (28)	0.0700 (42)
Laplace Approximation	0.179 (73)	0.00091 (91)	0.109 (91)
(Ryder et al., 2018)	0.140 (21)	0.00042 (42)	0.058 (11)
IEVI	**0.049** (95)	**0.00040** (88)	**0.015** (100)

4.5.2 Stochastic Volatility

The exponential Ornstein-Uhlenbeck model developed by Fouque and Sircar (2000), used to describe the behaviour of log price X_t and log volatility Z_t of a financial asset, is given by

$$\begin{aligned} \mathrm{d}X_t &= (\alpha - 0.5e^{2Z_t})\,\mathrm{d}t + e^{Z_t}\,\mathrm{d}B_t^X \\ \mathrm{d}Z_t &= -\gamma(Z_t - \mu)\,\mathrm{d}t + \sigma\,\mathrm{d}B_t^Z, \end{aligned} \tag{4.40}$$

where $\mathrm{cor}(\mathrm{d}B_t^X\,\mathrm{d}B_t^Z) = \rho$, and the latent state is defined as $\boldsymbol{X}_t = (X_t, Z_t)$. The parameters of interest are $\boldsymbol{\theta} = (\alpha, \log\gamma, \mu, \log\sigma, \log\frac{\rho+1}{1-\rho})$ for which we assume a flat prior on $\pi(\boldsymbol{\theta}) \propto 1$.

In this example, only the log price is observed according to the model

$$Y_i \overset{\text{ind}}{\sim} \mathrm{Normal}(X(t_i), \epsilon^2) \tag{4.41}$$

where $\epsilon = 0.001$ and $t_i = 0, \frac{10}{252}, \ldots, \frac{14990}{252}$ for a total of 1500 observations. For this experiment, we use a discretization resolution of $r = 2$, corresponding to a step size of $\Delta t = \frac{5}{252}$. The data is simulated using $\alpha = 0.04, \gamma = 3.0, \mu = -1.3, \sigma = 1.0$, and $\rho = -0.7$.

Due to the long observation sequence, training a RNN can result in high memory usage and long training times. To address this issue, Sutskever (2013) suggests splitting the sequence into shorter segments during training and updating the neural network parameters after each segment rather than after processing the entire sequence. This approach, known as truncated backpropagation through time (Sutskever, 2013), reduces total training time by performing more frequent parameter updates (Goodfellow et al., 2016). These updates are particularly beneficial in the early stages of training, when the parameters are randomly initialized, and even noisy gradient estimates can help the model begin moving toward the optimal solution.

To support this training technique, we adopt a different parameterization of the variational distribution than the one used in (4.39). In the earlier formulation, the variational parameters $(\boldsymbol{\mu}_{\boldsymbol{\phi}}, \boldsymbol{\Sigma}_{\boldsymbol{\phi}})$ are learned globally using the entire observation sequence, which must be processed end-to-end. To address this limitation, we instead use a RNN, parameterized by $\boldsymbol{\phi}_{\boldsymbol{\theta}}$, that can take any subsequence $\boldsymbol{Y}_{m_1:m_2}$ as input where $0 \leq m_1 \leq m_2 \leq M$ and output the mean and variance of the variational distribution,

$$(\boldsymbol{\mu}_{\boldsymbol{\theta}}, \boldsymbol{\Sigma}_{\boldsymbol{\theta}}^{1/2}) = \mathrm{RNN}(\boldsymbol{Y}_{m_1:m_2}; \boldsymbol{\phi}_{\boldsymbol{\theta}}). \tag{4.42}$$

We define the variational distribution as a multivariate normal with these outputs,

$$q(\boldsymbol{\theta} \mid \boldsymbol{Y}_{m_1:m_2}; \boldsymbol{\phi}_{\boldsymbol{\theta}}) = \mathrm{Normal}(\boldsymbol{\mu}_{\boldsymbol{\theta}}, \boldsymbol{\Sigma}_{\boldsymbol{\theta}}). \tag{4.43}$$

On the latent side, the variational distribution introduced in Section 4.3.1 already uses a RNN that can take subsequences of any length as input. Together with the RNN-based variational distribution for the SDE parameters, IEVI can be trained using batches extracted from the observation sequence. However, because these subsequences do not cover the entire observation sequence, the resulting variational distribution no longer approximates the smoothing distribution. To recover the smoothing behaviour, we adopt a two-phase training strategy: we first train using batched subsequences until convergence, and then fine-tune the model on the full observation sequence. This second phase typically requires only a few additional iterations. In this way, the fully trained variational distribution approximates the smoothing distribution while benefiting from a training procedure that scales efficiently with sequence length.

Figure 4.5 displays the convergence curves for maximizing the ELBO across several batch sizes of the observation sequence. Training the variational distribution using the full sequence (i.e., batch size of 1500) requires significantly more epochs to converge compared to smaller batch sizes. This difference is particularly pronounced at the beginning of training, where the more frequent parameter updates from smaller batches accelerate convergence. However, when the batch size is too small, ELBO maximization becomes unstable, reflecting well-documented challenges in neural network training (Goodfellow et al., 2016, Chapter 8).

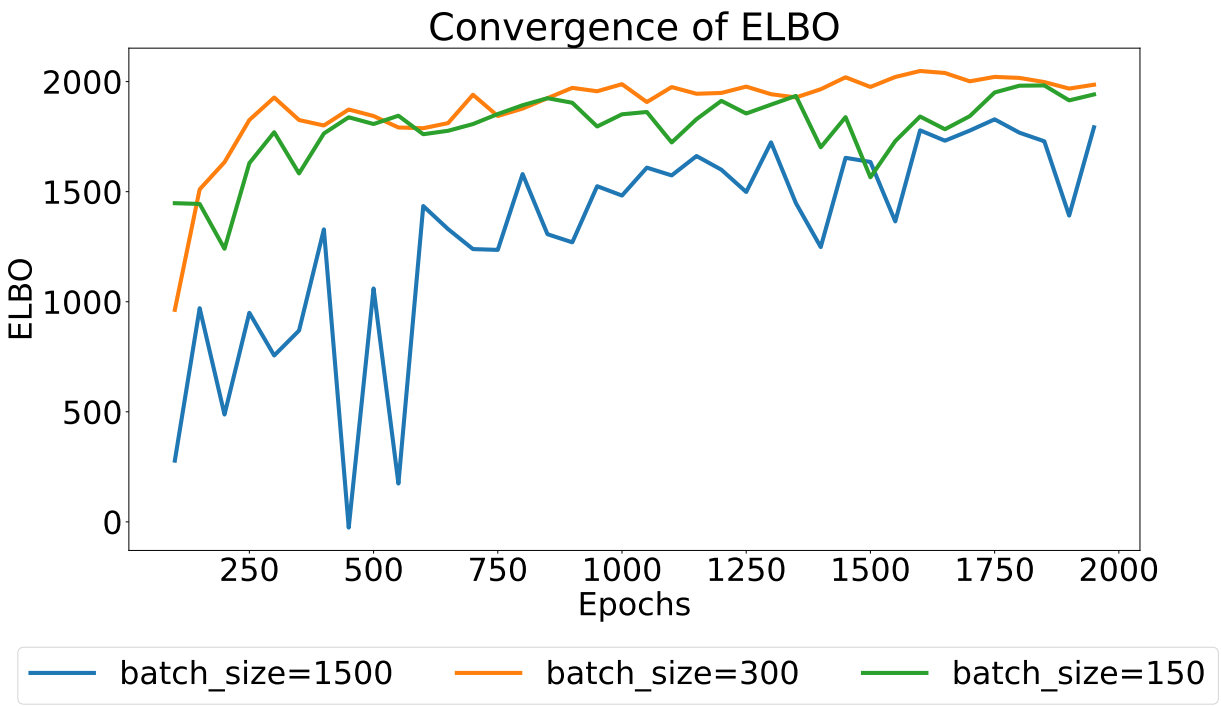

Figure 4.5: ELBO convergence curves for training IEVI using various batch sizes of the observation sequence.

Figure 4.6 displays $S = 1000$ samples of the latent variables $\boldsymbol{X}_{0:M}$ from the variational distributions produced by IEVI and the method of Ryder et al. (2018), together with samples obtained via MCMC approximating the joint posterior $p(\boldsymbol{\theta}, \boldsymbol{X}_{0:M} \mid \boldsymbol{Y}_{0:M})$ after burn-in. For the MCMC algorithm, we use the No-U-Turn Sampler (Hoffman and Gelman, 2014) as implemented in **BlackJAX**. MCMC parameters are tuned using the standard window adaptation algorithm. For the unobserved component $Z(t)$, the posterior of Ryder et al. (2018) fails to capture the ground truth trajectory. In contrast, the IEVI posterior closely matches that of MCMC and covers the true latent path.

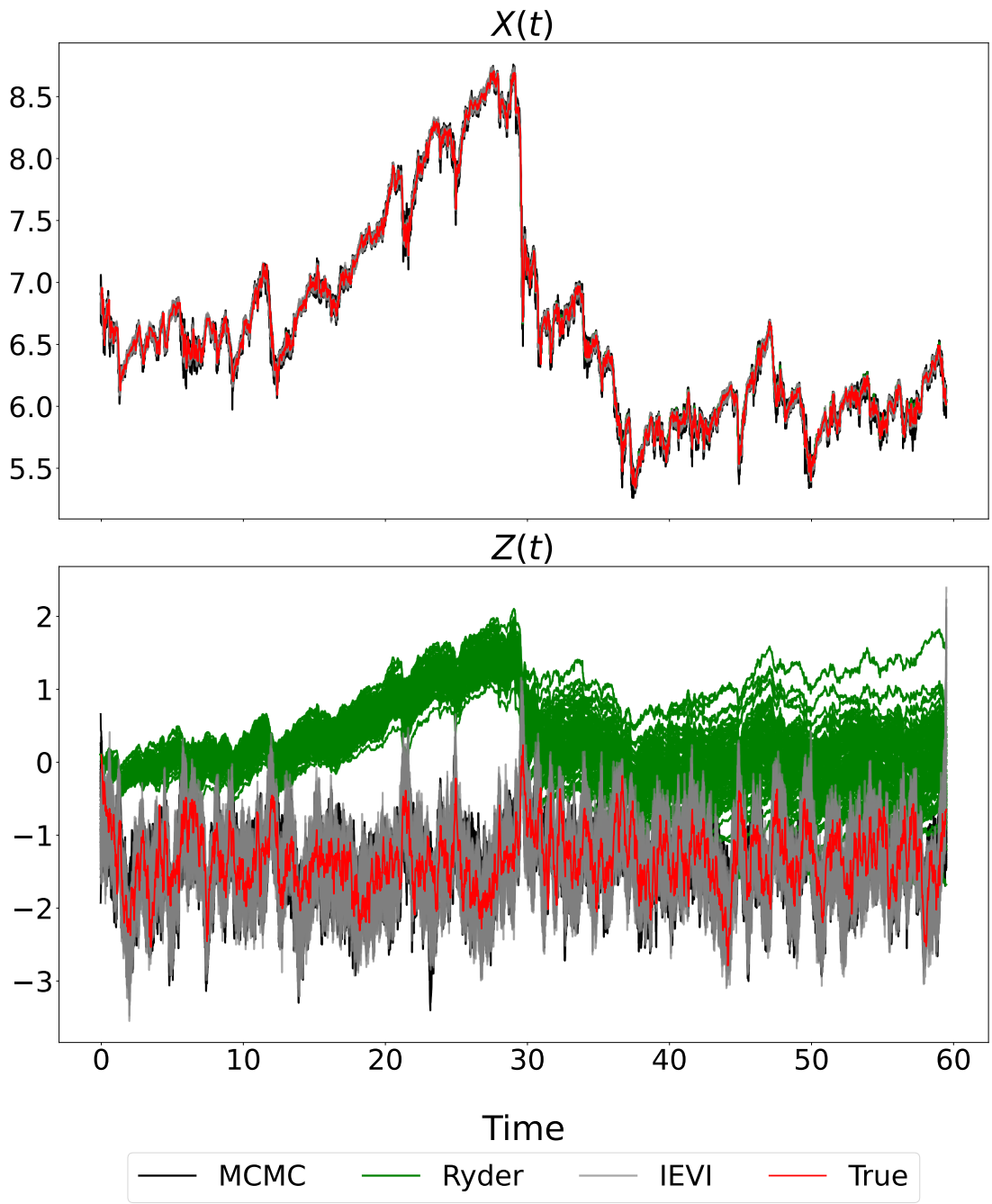

Figure 4.6: $S = 1000$ samples of the latent trajectories from the variational distributions produced by IEVI and the method of Ryder et al. (2018), as well as from MCMC post burn-in, shown alongside the ground truth latent trajectory and observed data.

Figure 4.7 presents the parameter posterior distributions estimated using the variational distributions of IEVI and the method of Ryder et al. (2018). For comparison, we also include results from two non-variational methods: particle filtering and the Laplace approximation. Additionally, we plot posterior samples of $\boldsymbol{\theta}$ from the MCMC approximation of $p(\boldsymbol{\theta}, \boldsymbol{X}_{0:M} \mid \boldsymbol{Y}_{0:M})$ after burn-in. Comparing only the variational methods, the posterior obtained by IEVI shows significantly better coverage of the true $\boldsymbol{\theta}$ than that of Ryder et al. (2018). However, both particle filtering and MCMC provide more accurate posteriors overall. This may be due in part to the long sequence length, which is known to pose challenges for variational methods (Turner and Sahani, 2011).

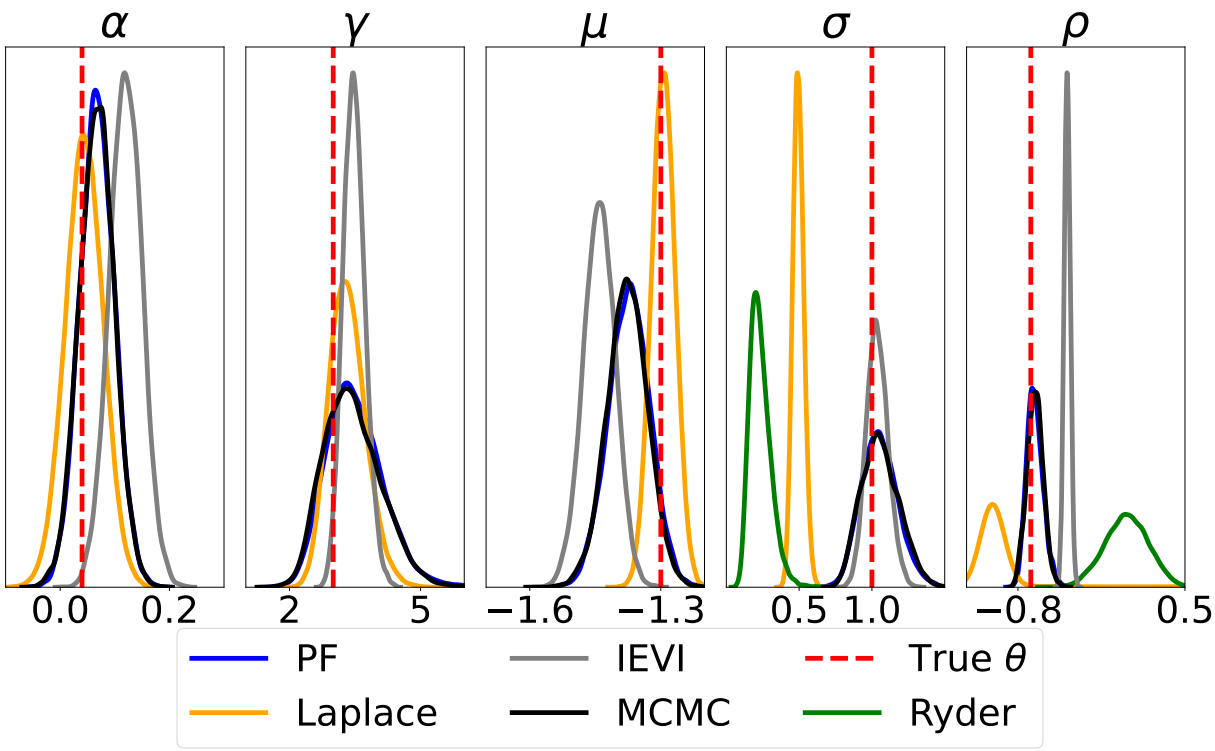

Figure 4.7: Parameter posteriors for the stochastic volatility model using various methods. Some of the Ryder et al. (2018) posteriors have been clipped for visual clarity. The corresponding mean (standard deviation) estimates for the missing Ryder et al. (2018) posteriors are: $\hat{\alpha} = 1.97(0.03), \hat{\gamma} = 0.17(0.04)$, and $\hat{\mu} = 3.43(0.05)$.

We further assess the accuracy and reliability of each method across 100 different

datasets. Table 4.2 reports the RMSE of the posterior mode for each method relative to the true parameter values. The number in brackets indicates how often the true parameter value was contained within the 95% credible interval across the 100 simulations. The results in Table 4.2 are consistent with those in Figure 4.7: MCMC and particle filtering give the lowest RMSE and the highest coverage.

Table 4.2: The RMSE for each parameter of the stochastic volatility model across 100 simulations, using various inference methods. The number in brackets indicates how many times the true parameter value was contained within the 95% credible interval.

Method	α	γ	μ	σ	ρ
MCMC	0.039 (90)	0.644 (86)	**0.046** (96)	0.125 (86)	0.067 (88)
Particle Filtering	0.039 (91)	**0.577** (91)	**0.046** (97)	**0.106** (93)	**0.060** (94)
Laplace	**0.038** (67)	0.952 (54)	0.087 (23)	0.552 (0)	0.299 (29)
(Ryder et al., 2018)	0.608 (22)	2.062 (1)	1.242 (1)	0.588 (0)	0.679 (0)
IEVI	0.308 (53)	1.441 (51)	0.571 (28)	0.441 (9)	0.212 (4)

4.5.3 Orange Tree Growth Model

The stochastic model proposed by Picchini and Ditlevsen (2011) and used by Whitaker et al. (2017), known as a stochastic differential mixed-effects model, describes the growth of individual orange trees in terms of their circumference (mm). It is given by

$$\mathrm{d}X_t^{(j)} = \frac{1}{\eta_1^{(j)}\eta_2^{(j)}} X_t^{(j)}(\eta_1^{(j)} - X_t^{(j)})\,\mathrm{d}t + \sigma\sqrt{X_t^{(j)}}\,\mathrm{d}B_t^{(j)}, \qquad j = 1, \ldots, D \tag{4.44}$$

where the random effects are distributed according to

$$\eta_1^{(j)} \overset{\text{iid}}{\sim} \text{Normal}(\eta_1, \sigma_{\eta_1}^2), \qquad \eta_2^{(j)} \overset{\text{iid}}{\sim} \text{Normal}(\eta_2, \sigma_{\eta_2}^2). \tag{4.45}$$

In this model, $\boldsymbol{\eta}^{(j)} = (\eta_1^{(j)}, \eta_2^{(j)})$ vary across trees, while the diffusion coefficient σ is a fixed effect shared among all trees. The parameters of interest are $\boldsymbol{\theta} = (\eta_1, \log\sigma_{\eta_1}, \eta_2, \log\sigma_{\eta_2}, \log\sigma)$, defined on the unconstrained scale. Following Whitaker et al. (2017), we use independent prior distributions as follows: η_1 and η_2 follow $\text{Normal}(0, 100^2)$ priors, while $1/\sigma_{\eta_1}^2, 1/\sigma_{\eta_2}^2$ and $1/\sigma^2$ follow $\text{Gamma}(1, 0.01)$ priors.

The observation model is specified by

$$Y_i^{(j)} \overset{\text{ind}}{\sim} \text{Normal}(X(t_i)^{(j)}, \boldsymbol{\epsilon}^2), \qquad j = 1, \ldots, D \tag{4.46}$$

where $\boldsymbol{\epsilon} = 1$ and $t_i = 0, 200, 400, \ldots, 3000$. For this experiment, we use a discretization resolution of $r = 2$, corresponding to a step size of $\Delta t = 100$.

We first examine a simple setting where $D = 1$ (i.e., a single tree) and evaluate the quality of the approximation to $p(\boldsymbol{X}_{0:M} \mid \boldsymbol{Y}_{0:M}, \boldsymbol{\eta}, \boldsymbol{\theta})$ assuming that both $\boldsymbol{\eta}$ and $\boldsymbol{\theta}$ are known. Specifically, we compare the effectiveness of our IEVI method against that of the method of Ryder et al. (2018) and particle filtering. These approximations are evaluated relative to the true latent process, simulated using $(\eta_1, \sigma_{\eta_1}, \eta_2, \sigma_{\eta_2}, \sigma) = (195, 25, 350, 52.5, 0.08)$ and initial value $X_0 = 30$.

Figure 4.8 displays $S = 1000$ sample drawn from the variational distributions produced by IEVI and the method of Ryder et al. (2018). For particle filtering, we use $P = 10,000$ particles with a multinomial resampler, combined with the standard particle smoother (Kitagawa, 1996; Doucet and Johansen, 2009) which produces an accurate estimate of the true posterior $p(\boldsymbol{X}_{0:M} \mid \boldsymbol{Y}_{0:M}, \boldsymbol{\eta}, \boldsymbol{\theta})$. Also shown are the ground truth latent trajectory, the observed data, and 1000 MCMC samples post burn-in. For the MCMC algorithm, we use HMC (Duane et al., 1987) with 10 leapfrog steps, with the remaining parameters tuned using the window adaptation algorithm (Stan Development Team, 2023). MCMC, particle filtering, and IEVI produce posterior distributions that provide good coverage of the true latent trajectory. Moreover, all three methods yield highly similar posteriors. In contrast, the variational distribution produced by the method of Ryder et al. (2018) fails to capture the true trajectory at several time points.

We now assume there are $D = 100$ trees, each with independent random effects $(\eta_1^{(j)}, \eta_2^{(j)})$ for $j = 1, \ldots, D$. In this setting, we must approximate three distributions: the latent distribution, the random effects distribution, and the parameter distribution. As such, we cannot directly use the variational model from (4.24). Instead, we adopt a similar model,

$$q(\boldsymbol{\theta}, \boldsymbol{\eta}^{(1:D)}, \boldsymbol{X}_{0:M}^{(1:D)} \mid \boldsymbol{Y}_{0:M}^{(1:D)}; \boldsymbol{\phi}) = q(\boldsymbol{\theta} \mid \boldsymbol{Y}_{0:M}^{(1:D)}; \boldsymbol{\phi}_{\boldsymbol{\theta}}) \prod_{j=1}^{D} q(\boldsymbol{\eta}^{(j)} \mid \boldsymbol{Y}_{0:M}^{(j)}, \boldsymbol{\theta}; \boldsymbol{\phi}_{\boldsymbol{\eta}}) q(\boldsymbol{X}_{0:M}^{(j)} \mid \boldsymbol{Y}_{0:M}^{(j)}, \boldsymbol{\eta}^{(j)}, \boldsymbol{\theta}; \boldsymbol{\phi}_{\boldsymbol{X}}) \tag{4.47}$$

where $\boldsymbol{\eta}^{(1:D)} = (\eta_1^{(1)}, \eta_2^{(1)}, \ldots, \eta_1^{(D)}), \eta_2^{(D)}$ denotes the collection of random effects across all trees, $\boldsymbol{Y}_{0:M}^{(1:D)} = (\boldsymbol{Y}_{0:M}^{(1)}, \ldots \boldsymbol{Y}_{0:M}^{(D)})$ are the corresponding observations, and $\boldsymbol{X}_{0:M}^{(1:D)}$ are the latent trajectories.

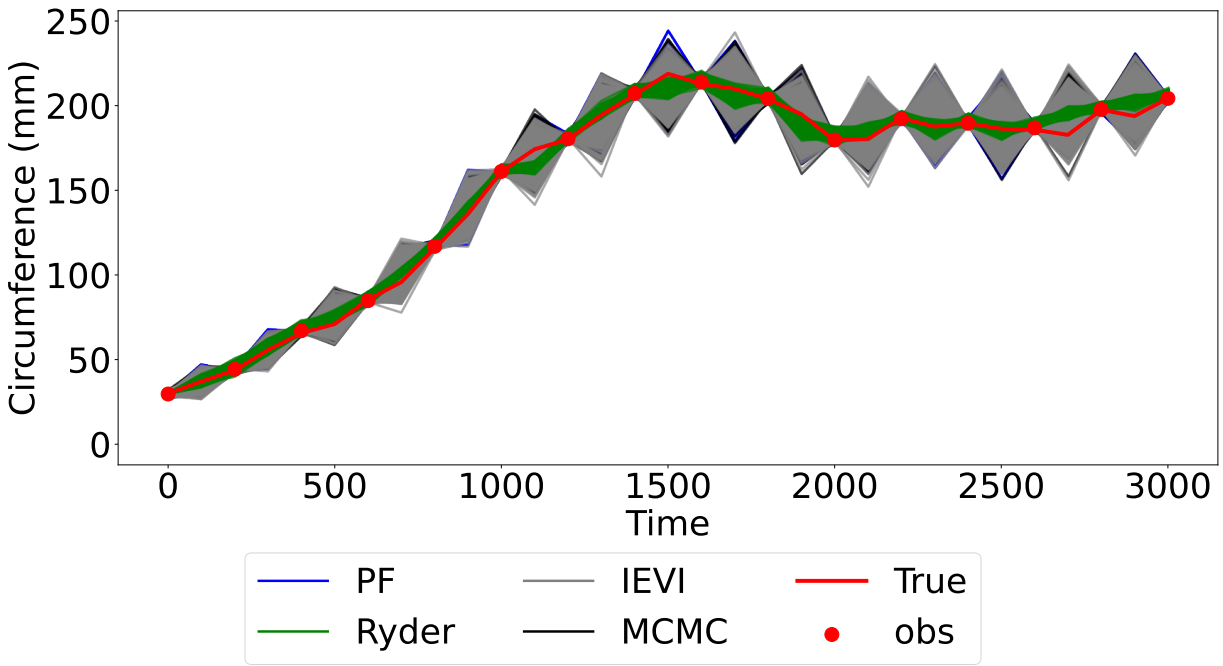

Figure 4.8: $S = 1000$ sample draws from the variational distributions of IEVI and the method of Ryder et al. (2018), and $P = 10,000$ particles from particle filtering, shown alongside the ground truth latent trajectory and observed data for the orange tree growth model. Also included are 1000 MCMC samples post burn-in.

For the parameter distribution, we use the simple multivariate formulation of (4.39) and the standard variational distributions for the latent variables described in Section 4.3. Note that the same RNN, parameterized by $\boldsymbol{\phi_X}$, is shared across all trees, and is trained jointly over the entire dataset. That is, we model a single variational distribution $q(\boldsymbol{X}_{0:M} \mid \boldsymbol{Y}_{0:M}, \boldsymbol{\eta}, \boldsymbol{\theta}; \boldsymbol{\phi_X})$ conditioned on the observations and random effects of each tree, rather than learning separate distributions for each tree.

For the variational distribution over random effects, we use a conditional multivariate normal, where the mean and variance are parameterized by a RNN,

$$\boldsymbol{\mu}_\eta, \boldsymbol{\Sigma}_\eta^{1/2} = \mathrm{RNN}(\boldsymbol{Y}_{0:M}, \boldsymbol{\theta}; \boldsymbol{\phi}_\eta). \tag{4.48}$$

A single RNN with parameters $\boldsymbol{\phi}_\eta$ is trained across all trees. For each tree j, the RNN is evaluated on the tree-specific observation sequence $\boldsymbol{Y}_{0:M}^{(j)}$ to produce the corresponding

parameters $\boldsymbol{\mu}_\eta^{(j)}$ and $(\boldsymbol{\Sigma}_\eta^{1/2})^{(j)}$ for the variational distribution. To sample from $q(\boldsymbol{\eta}^{(j)} \mid \boldsymbol{Y}_{0:M}^{(j)}, \boldsymbol{\theta}; \boldsymbol{\phi}_{\boldsymbol{\eta}})$, we apply the reparameterization trick,

$$\boldsymbol{\eta}^{(j,i)} = \boldsymbol{\mu}_\eta^{(j)} + (\boldsymbol{\Sigma}_\eta^{1/2})^{(j)} \boldsymbol{\epsilon}_\eta^{(j,i)} \tag{4.49}$$

where $\boldsymbol{\epsilon}_\eta^{j,i)} \overset{\text{ind}}{\sim} \text{Normal}(\mathbf{0}, \boldsymbol{I}_2)$ for $i = 1, \ldots, S$ and $j = 1, \ldots, D$. Formulating the two RNNs in this way allows batching over the number of trees during training, enabling the method to scale efficiently to large D. This is because RNN parameter updates can be performed more frequently, as reliable gradient estimates can be obtained without using the full dataset (Goodfellow et al., 2016).

While the variational distribution $q(\boldsymbol{\theta} \mid \boldsymbol{Y}_{0:M}^{(1:D)}; \boldsymbol{\phi}_{\boldsymbol{\theta}})$ can serve as a reasonable approximation to the parameter posterior, it tends to undercover the true parameters in the 95% credible interval, which is a well-known limitation of variational inference (Blei et al., 2017). On the other hand, MCMC methods provide better coverage of the true parameters, but are computationally expensive, particularly when sampling over high dimensional latent variables. As shown previously in Figure 4.8, the variational distribution over the latent states, $q(\boldsymbol{X}_{0:M} \mid \boldsymbol{Y}_{0:M}, \boldsymbol{\eta}, \boldsymbol{\theta}; \boldsymbol{\phi}_{\boldsymbol{X}})$, provides a good approximation to the true posterior $p(\boldsymbol{X}_{0:M} \mid \boldsymbol{Y}_{0:M}, \boldsymbol{\eta}, \boldsymbol{\theta})$. This motivates a hybrid approach that combines MCMC with variational inference.

We start with the identity:

$$\begin{aligned} p(\boldsymbol{\theta}, \boldsymbol{\eta}^{(1:D)} \mid \boldsymbol{Y}_{0:M}^{(1:D)}) &\propto \frac{p(\boldsymbol{\theta}, \boldsymbol{\eta}^{(1:D)}, \boldsymbol{X}_{0:M}^{(1:D)}, \boldsymbol{Y}_{0:M}^{(1:D)})}{p(\boldsymbol{X}_{0:M}^{(1:D)} \mid \boldsymbol{Y}_{0:M}^{(1:D)}, \boldsymbol{\eta}^{(1:D)}, \boldsymbol{\theta})} \\ &= \frac{\pi(\boldsymbol{\theta}) p(\boldsymbol{\eta}^{(1:D)} \mid \boldsymbol{\theta}) p(\boldsymbol{X}_{0:M}^{(1:D)} \mid \boldsymbol{\eta}^{(1:D)}, \boldsymbol{\theta}) p(\boldsymbol{Y}_{0:M}^{(1:D)} \mid \boldsymbol{X}_{0:M}^{(1:D)}, \boldsymbol{\eta}^{(1:D)}, \boldsymbol{\theta})}{p(\boldsymbol{X}_{0:M}^{(1:D)} \mid \boldsymbol{Y}_{0:M}^{(1:D)}, \boldsymbol{\eta}^{(1:D)}, \boldsymbol{\theta})}, \end{aligned} \tag{4.50}$$

where the numerator can be computed directly from the generative model, but the denominator is intractable. To approximate the posterior in (4.50) we replace the denominator with the variational distribution $q(\boldsymbol{X}_{0:M}^{(1:D)} \mid \boldsymbol{Y}_{0:M}^{(1:D)}, \boldsymbol{\eta}^{(1:D)}, \boldsymbol{\theta}; \hat{\boldsymbol{\phi}}_{\boldsymbol{X}})$, where $\hat{\boldsymbol{\phi}}_{\boldsymbol{X}}$ is the optimal variational parameter learned during training of (4.47). We then perform MCMC on this approximate posterior. During MCMC, the variational parameters are held fixed and do not need to be updated. Thus, both sampling and log-density evaluation of the variational distribution are computationally efficient.

Normally, MCMC would require sampling over all variables, including the latent variables $\boldsymbol{X}_{0:M}^{(1:D)}$. By approximating the conditional distribution with the variational distribution, we avoid explicitly sampling the latent variables. This substantially reduces the

computational burden, as MCMC now operates only on the lower dimensional space of $\boldsymbol{\theta}$ and $\boldsymbol{\eta}^{(1:D)}$. For the MCMC algorithm, we use HMC with 10 leapfrog steps, implemented via the **BlackJAX** library (Lao and Louf, 2020). The remaining HMC parameters are tuned using the window adaptation algorithm.

Although particle filtering is one of the baseline methods considered, directly applying it to estimate the likelihood function $\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\eta}^{(1:D)} \mid \boldsymbol{Y}_{0:M}^{(1:D)})$ yields degenerate results, where the marginal posteriors of σ_{η_1} and σ_{η_2} are sharply peaked near zero. This behaviour is likely due to identifiability issues in the model (Whitaker et al., 2017), exacerbated by inaccuracies in the particle filtering estimate of the likelihood. To address this, we apply the Max-and-Smooth approach proposed by Hrafnkelsson et al. (2021) to construct an approximate likelihood $\hat{\mathcal{L}}(\boldsymbol{\theta} \mid \boldsymbol{Y}_{0:M}^{(1:D)})$.

Max-and-Smooth mitigates this issue by first approximating the per-tree likelihood $\hat{\mathcal{L}}(\boldsymbol{\eta}^i, \sigma \mid \boldsymbol{Y}_{0:M}^i)$ independently for each tree $i = 1, \ldots, D$, before combining them in the Smooth step. Since the true per-tree likelihood, $\mathcal{L}(\boldsymbol{\eta}^i, \sigma \mid \boldsymbol{Y}_{0:M}^i)$, is intractable, we use particle filtering to obtain an approximation. Max-and-Smooth then fits a multivariate normal distribution to this particle filtering–based approximation. This localized estimation avoids the identifiability issues that arise when estimating all parameters jointly, and helps prevent the marginal posterior of σ_{η_1} and σ_{η_2} from collapsing to zero. We refer to this approach as the Max-and-Smooth particle filter.

Table 4.3 displays the RMSE of the posterior mode for each method relative to the true parameter values across 100 simulated datasets. The number in brackets indicates how many times the true parameter value was contained within the 95% credible interval across the 100 simulations. On its own, IEVI achieves low RMSE for many of the parameters compared to the Max-and-Smooth particle filter and the Laplace approximation. However, its coverage is poor, providing empirical evidence that variational inference tends to underestimate posterior variance (Blei et al., 2017). In contrast, the hybrid approach that combines IEVI with MCMC achieves substantially better coverage while maintaining low RMSE.

Table 4.3: The RMSE for each parameter of the orange tree growth model across 100 simulations, using various inference methods. The number in brackets indicates how many times the true parameter value was contained within the 95% credible interval.

Method	ϕ_1	σ_{ϕ_1}	ϕ_2	σ_{ϕ_2}	σ
Max-Smooth PF	3.269 (85)	3.179 (73)	50.51 (0)	**5.817** (95)	0.005 (39)
Laplace	3.464 (92)	3.682 (84)	17.47 (93)	43.31 (28)	0.014 (43)
IEVI	3.627 (7)	5.614 (73)	6.792 (66)	19.67 (77)	0.042 (8)
IEVI MCMC	**2.996** (90)	**2.199** (95)	**5.732** (92)	8.870 (79)	**0.002** (79)

We now scale the problem from $N = 100$ to $N = 1000$ trees, at which point MCMC over all variables, $(\boldsymbol{X}_{0:M}^{(1:D)}, \boldsymbol{\eta}^{(1:D)}, \boldsymbol{\theta})$, becomes computationally infeasible. In contrast, the training time of IEVI increases only slightly. Moreover, sampling from the MCMC-enhanced IEVI remains efficient, as the dimension of $\boldsymbol{\eta}$ is only 2 per tree. Figure 4.9 displays the parameter posteriors for the Max-and-Smooth particle filter, the Laplace approximation and three variants of IEVI. These include the base method, the MCMC-enhanced version, and the Max-and-Smooth version.

To apply Max-and-Smooth with IEVI, we approximate the intractable per-tree likelihood

$$\begin{aligned}\mathcal{L}(\boldsymbol{\eta}^i, \sigma \mid \boldsymbol{Y}_{0:M}^i) &= \frac{p(\boldsymbol{X}_{0:M}^i, \boldsymbol{Y}_{0:M}^i \mid \boldsymbol{\eta}^i, \sigma)}{p(\boldsymbol{X}_{0:M}^i \mid \boldsymbol{Y}_{0:M}^i, \boldsymbol{\eta}^i, \sigma)} \\ &= \frac{p(\boldsymbol{X}_{0:M}^i \mid \boldsymbol{\eta}^i, \sigma) p(\boldsymbol{Y}_{0:M}^i \mid \boldsymbol{X}_{0:M}^i, \boldsymbol{\eta}^i, \sigma)}{p(\boldsymbol{X}_{0:M}^i \mid \boldsymbol{Y}_{0:M}^i, \boldsymbol{\eta}^i, \sigma)}.\end{aligned} \tag{4.51}$$

We approximate this likelihood by drawing samples from the trained IEVI variational distribution $q(\boldsymbol{X}_{0:M} \mid \boldsymbol{Y}_{0:M}, \boldsymbol{\eta}, \boldsymbol{\theta}; \hat{\boldsymbol{\phi}}_{\boldsymbol{X}})$, evaluating the numerator of (4.51) at these samples, and approximating the intractable denominator with the corresponding variational density. Max-and-Smooth is then applied in the usual way. The only difference from the particle filtering version of Max-and-Smooth lies in how the per-tree likelihood is approximated: here, IEVI is used in place of particle filtering.

The posteriors obtained from the Laplace approximation fail to cover any of the true parameters, potentially due to the difficulty of integrating out a large number of latent variables. The two Max-and-Smooth methods show similar behaviour, producing posteriors with very low variance and failing to capture the true value of σ. Among all methods, the

hybrid IEVI MCMC approach achieves the highest coverage. In particular, compared to pure IEVI, it produces posteriors with higher variance.

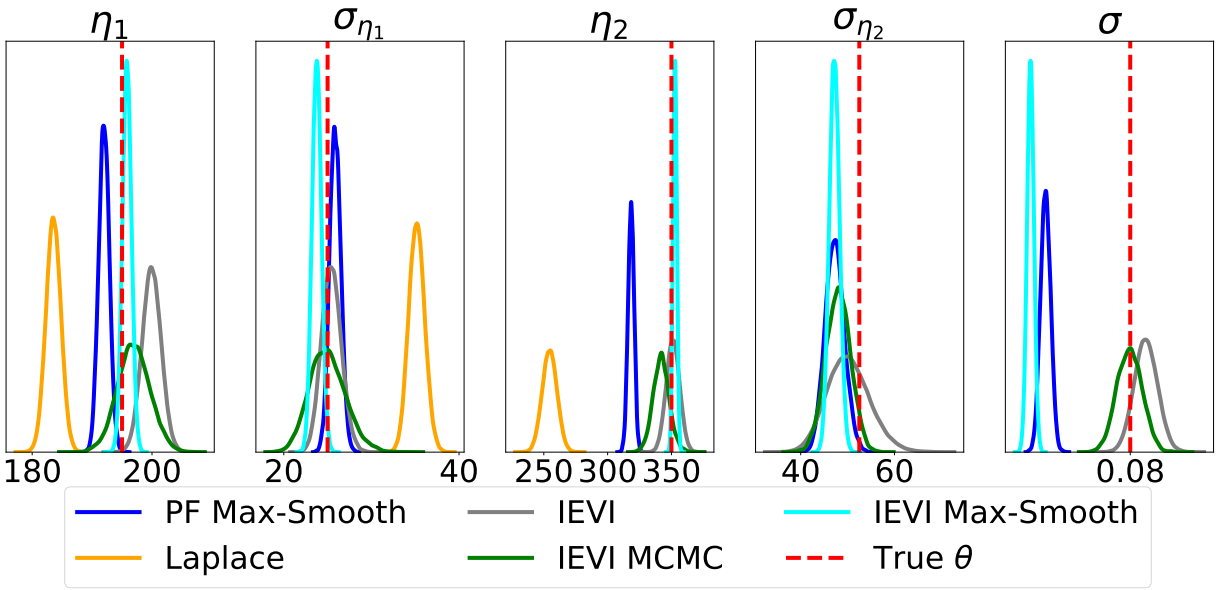

Figure 4.9: Parameter posteriors for the orange tree growth model using various methods. Some of the Laplace approximation posteriors have been clipped for visual clarity. The corresponding mean (standard deviation) estimates for the missing Laplace posteriors are: $\hat{\sigma}_{\eta_2} = 144(4.31)$, and $\hat{\sigma} = 0.18(0.0012)$.

4.6 Discussion

This chapter introduced IEVI, a scalable and flexible variational inference framework for parameter inference in SDE models. It transforms the intractable integration over high-dimensional latent variables into a tractable optimization problem. By combining modern advances in optimizing RNNs with the Kalman recursions, IEVI approximates the latent smoothing distribution in linear time with respect to the number of discretization steps. Owing to its flexibility, the framework supports training regimes that scale efficiently with both the number of SDE processes and the number of time steps per process, as well as a variety of parameter posterior approximation strategies.

Although IEVI outperforms existing variational methods in several experiments, it performs worse than particle filtering in the stochastic volatility problem, which involves a very long observation sequence. This is partly due to the inherent bias in variational approximations, which can accumulate over time (Turner and Sahani, 2011). Addressing this limitation remains an open area for future research.

Currently, IEVI assumes that the latent posteriors can be well approximated by a Gaussian distribution. While this allows for efficient inference, it may be a poor approximation in problems where the true posterior is multi-modal. A promising direction for future work is to explore richer variational families, such as mixtures of Gaussians, which may better capture complex posterior structures.

Chapter 5

Conclusion

The purpose of this thesis was to develop several methods for parameter inference in models that use ordinary and stochastic differential equations. The main focus was on designing efficient approximate inference algorithms that are more practical than traditional MCMC methods, especially for large-scale problems. For ODE models, we proposed novel approaches in probabilistic numerics that can approximate the ODE likelihood function at low computational cost. For SDE models, we introduced a flexible variational inference framework that efficiently approximates the parameter posterior under Bayesian inference.

In Chapter 2, we investigated probabilistic ODE solvers based on the Bayesian filtering paradigm. We proved that solvers that use the RTS smoother converge to the true solution at the same rate as those using the Kalman filter. These theoretical guarantees ensure that such solvers can approximate the true solution arbitrarily well as more computational resources are used. Our experiments further demonstrated that, although the convergence rates are the same, the RTS smoother produces more accurate estimates than the Kalman filter alone. We also compared several parameter inference methods for ODE models using probabilistic solvers, including a novel and efficient variant of MAGI. The results showed that, at sufficiently small step sizes, the parameter posteriors from each probabilistic method successfully covered the true parameter values.

Chapter 3 introduced DALTON, a data-adaptive probabilistic ODE likelihood approximation that uses the Bayesian filtering model. DALTON leverages information from noisy observations to directly guide the ODE solution process, which reduces parameter sensitivity in the likelihood. It is also highly flexible, supporting ODE systems with partially unobserved components and accommodating observation noise models with arbitrary distributions. Finally, DALTON is computationally efficient, scaling linearly with both the

number of ODE variables and the number of discretization time steps. Our results showed that DALTON is particularly effective in a chaotic system with high parameter sensitivity, where it produced more robust parameter estimates than even those obtained using the true ODE likelihood.

In Chapter 4, we introduced IEVI, a variational inference method for estimating parameters in SDE models using Bayesian inference. IEVI effectively captures information propagation in SSMs by approximating the smoothing distribution using a RNN that mimics the forward-backward algorithm. Through several experiments, we demonstrated that IEVI achieves sublinear training complexity with respect to both the length of observation sequences and the number of datasets. Notably, in a mixed-effects setting involving thousands of SDE processes, where traditional MCMC methods were computationally infeasible, IEVI produced more reliable parameter posteriors than existing variational and non-variational methods.

References

Christophe Andrieu and Arnaud Doucet. Particle filtering for partially observed gaussian state space models. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 64(4):827–836, 10 2002. ISSN 1369-7412. doi: 10.1111/1467-9868.00363. URL https://doi.org/10.1111/1467-9868.00363.

Evan Archer, Il Memming Park, Lars Buesing, John Cunningham, and Liam Paninski. Black box variational inference for state space models, 2015.

Kendall Atkinson, Weimin Han, and David E Stewart. *Numerical Solution of Ordinary Differential Equations*. John Wiley & Sons, 2009. ISBN 978-0-470-04294-6.

David Barber and Yali Wang. Gaussian processes for Bayesian estimation in ordinary differential equations. In Eric P. Xing and Tony Jebara, editors, *Proceedings of the 31st international conference on machine learning*, volume 32 of *Proceedings of machine learning research*, pages 1485–1493, Bejing, China, 2014. PMLR. URL https://proceedings.mlr.press/v32/barber14.html.

Philipp Becker, Harit Pandya, Gregor Gebhardt, Cheng Zhao, C. James Taylor, and Gerhard Neumann. Recurrent Kalman Networks: Factorized Inference in High-Dimensional Deep Feature Spaces. In *Proceedings of the 36th International Conference on Machine Learning*, pages 544–552. PMLR, May 2019. URL https://proceedings.mlr.press/v97/becker19a.html. ISSN: 2640-3498.

Thomas Bengtsson, Peter Bickel, and Bo Li. Curse-of-dimensionality revisited: Collapse of the particle filter in very large scale systems. *IMS Collections*, 2, 06 2008. doi: 10.1214/193940307000000518.

David M. Blei, Alp Kucukelbir, and Jon D. McAuliffe and. Variational inference: A review for statisticians. *Journal of the American Statistical Association*, 112(518):859–877, 2017. doi: 10.1080/01621459.2017.1285773. URL https://doi.org/10.1080/01621459.2017.1285773.

Mathieu Blondel, Quentin Berthet, Marco Cuturi, Roy Frostig, Stephan Hoyer, Felipe Llinares-López, Fabian Pedregosa, and Jean-Philippe Vert. Efficient and modular implicit differentiation. *arXiv preprint arXiv:2105.15183*, 2021.

Nathanael Bosch, Philipp Hennig, and Filip Tronarp. Calibrated adaptive probabilistic ODE solvers. In Arindam Banerjee and Kenji Fukumizu, editors, *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, volume 130 of *Proceedings of Machine Learning Research*, pages 3466–3474. PMLR, 13–15 Apr 2021. URL https://proceedings.mlr.press/v130/bosch21a.html.

Nathanael Bosch, Filip Tronarp, and Philipp Hennig. Pick-and-mix information operators for probabilistic ODE solvers. In Gustau Camps-Valls, Francisco J. R. Ruiz, and Isabel Valera, editors, *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*, volume 151 of *Proceedings of Machine Learning Research*, pages 10015–10027. PMLR, 28–30 Mar 2022. URL https://proceedings.mlr.press/v151/bosch22a.html.

James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018. URL http://github.com/google/jax.

John Charles Butcher. *Numerical Methods for Ordinary Differential Equations*. John Wiley & Sons, 2008.

Ben Calderhead, Mark Girolami, and Neil D Lawrence. Accelerating Bayesian inference over nonlinear differential equations with Gaussian processes. In *Advances in neural information processing systems*, pages 217–224, 2009.

J. Cao, L. Wang, and J. Xu. Robust estimation for ordinary differential equation models. *Biometrics*, 67(4):1305–1313, 2011. doi: https://doi.org/10.1111/j.1541-0420.2011.01577.x. URL https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1541-0420.2011.01577.x.

Oksana A. Chkrebtii, David A. Campbell, Ben Calderhead, and Mark A. Girolami. Bayesian solution uncertainty quantification for differential equations. *Bayesian Analysis*, 11(4):1239–1267, December 2016. ISSN 1936-0975. doi: 10.1214/16-BA1017. URL https://projecteuclid.org/euclid.ba/1473276259.

Patrick R. Conrad, Mark Girolami, Simo Särkkä, Andrew Stuart, and Konstantinos Zygalakis. Statistical analysis of differential equations: introducing probability measures

on numerical solutions. *Statistics and Computing*, 27(4):1065–1082, July 2017. ISSN 0960-3174, 1573-1375. doi: 10.1007/s11222-016-9671-0. URL http://link.springer.com/10.1007/s11222-016-9671-0.

Adrien Corenflos, James Thornton, George Deligiannidis, and Arnaud Doucet. Differentiable particle filtering via entropy-regularized optimal transport. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 2100–2111. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/corenflos21a.html.

Sarat C. Dass, Jaeyong Lee, Kyoungjae Lee, and Jonghun Park. Laplace based approximate posterior inference for differential equation models. *Statistics and Computing*, 27(3):679–698, may 2017. ISSN 0960-3174. doi: 10.1007/s11222-016-9647-0.

DeepMind, Igor Babuschkin, Kate Baumli, Alison Bell, Surya Bhupatiraju, Jake Bruce, Peter Buchlovsky, David Budden, Trevor Cai, Aidan Clark, Ivo Danihelka, Antoine Dedieu, Claudio Fantacci, Jonathan Godwin, Chris Jones, Ross Hemsley, Tom Hennigan, Matteo Hessel, Shaobo Hou, Steven Kapturowski, Thomas Keck, Iurii Kemaev, Michael King, Markus Kunesch, Lena Martens, Hamza Merzic, Vladimir Mikulik, Tamara Norman, George Papamakarios, John Quan, Roman Ring, Francisco Ruiz, Alvaro Sanchez, Laurent Sartran, Rosalia Schneider, Eren Sezener, Stephen Spencer, Srivatsan Srinivasan, Miloš Stanojević, Wojciech Stokowiec, Luyu Wang, Guangyao Zhou, and Fabio Viola. The DeepMind JAX Ecosystem, 2020. URL http://github.com/google-deepmind.

P Diaconis. Bayesian numerical analysis. In J Berger and S Gupta, editors, *Statistical Decision Theory and Related Topics IV*, volume 1, pages 163–175. Springer-Verlag, New York, 1988.

Frank Dondelinger, Dirk Husmeier, Simon Rogers, and Maurizio Filippone. ODE parameter inference using adaptive gradient matching with Gaussian processes. In Carlos M. Carvalho and Pradeep Ravikumar, editors, *Proceedings of the sixteenth international conference on artificial intelligence and statistics*, volume 31 of *Proceedings of machine learning research*, pages 216–228, Scottsdale, Arizona, USA, 2013. PMLR. URL https://proceedings.mlr.press/v31/dondelinger13a.html.

John R Dormand and Peter J Prince. A family of embedded Runge-Kutta formulae. *Journal of Computational and Applied Mathematics*, 6(1):19–26, 1980. Publisher: Elsevier.

Randal Douc, Olivier Cappé, and Eric Moulines. Comparison of resampling schemes for particle filtering, 2005. URL https://arxiv.org/abs/cs/0507025.

Arnaud Doucet and Adam Johansen. A tutorial on particle filtering and smoothing: Fifteen years later. *Handbook of Nonlinear Filtering*, 12, 01 2009.

Simon Duane, A. D. Kennedy, Brian J. Pendleton, and Duncan Roweth. Hybrid Monte Carlo. *Physics Letters B*, 195(2):216–222, September 1987. doi: 10.1016/0370-2693(87)91197-X.

Jean-Pierre Fouque and K Sircar. *Derivatives in Financial Markets with Stochastic Volatility*. Cambridge University Press, 01 2000.

Weber Frank, Theers Stefan, Surmann Dirk, Ligges Uwe, and Weihs Claus. Sensitivity analysis of ordinary differential equation models. Technical report, Technische Universität Dortmund, 2018.

Andrew Gelman, John B Carlin, Hal S Stern, David B Dunson, Aki Vehtari, and Donald B Rubin. *Bayesian Data Analysis*. Chapman & Hall, New York, NY, 3rd edition, 2013. ISBN 978-0-429-11307-9.

Sanmitra Ghosh, Srinandan Dasmahapatra, and Koushik Maharatna. Fast approximate Bayesian computation for estimating parameters in differential equations. *Statistics and Computing*, 27(1):19–38, 2017. ISSN 0960-3174, 1573-1375. doi: 10.1007/s11222-016-9643-4. URL http://link.springer.com/10.1007/s11222-016-9643-4.

Sanmitra Ghosh, Paul J. Birrell, and Daniela De Angelis. Differentiable Bayesian inference of SDE parameters using a pathwise series expansion of Brownian motion. In *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*, pages 10982–10998. PMLR, May 2022. URL https://proceedings.mlr.press/v151/ghosh22a.html. ISSN: 2640-3498.

Nikolaos Gianniotis. Mixed variational inference. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, Budapest, Hungary, 2019. IEEE. ISBN 978-1-72811-985-4. doi: 10.1109/IJCNN.2019.8852348. URL https://ieeexplore.ieee.org/document/8852348/.

A. Golightly and D.J. Wilkinson. Bayesian inference for nonlinear multivariate diffusion models observed with error. *Computational Statistics & Data Analysis*, 52(3):1674–1693, 2008. ISSN 0167-9473. doi: https://doi.org/10.1016/j.csda.2007.05.019. URL https://www.sciencedirect.com/science/article/pii/S0167947307002198.

Andrew Golightly and Darren J. Wilkinson. Bayesian parameter inference for stochastic biochemical network models using particle markov chain monte carlo. *Interface Focus*, 1(6):807–820, 2011. doi: 10.1098/rsfs.2011.0047. URL https://royalsocietypublishing.org/doi/abs/10.1098/rsfs.2011.0047.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

Nico S. Gorbach, Stefan Bauer, and Joachim M. Buhmann. Scalable variational inference for dynamical systems. In *Proceedings of the 31st international conference on neural information processing systems*, NIPS'17, pages 4809–4818, Red Hook, NY, USA, 2017. Curran Associates Inc. ISBN 978-1-5108-6096-4.

David F Griffiths and Desmond J Higham. *Numerical Methods for Ordinary Differential Equations: Initial Value Problems*. Springer Science & Business Media, 2010.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 1026–1034, 2015. doi: 10.1109/ICCV.2015.123.

Philipp Hennig, Michael A. Osborne, and Mark Girolami. Probabilistic numerics and uncertainty in computations. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 471(2179):20150142, July 2015. ISSN 1364-5021, 1471-2946. doi: 10.1098/rspa.2015.0142. URL https://royalsocietypublishing.org/doi/10.1098/rspa.2015.0142.

A Hindmarsh. Odepack. a collection of ode system solvers. *Scientific Computing*, 1 1992. URL https://www.osti.gov/biblio/145724.

Matthew D. Hoffman and Andrew Gelman. The no-u-turn sampler: Adaptively setting path lengths in hamiltonian monte carlo. *Journal of Machine Learning Research*, 15(47): 1593–1623, 2014. URL http://jmlr.org/papers/v15/hoffman14a.html.

Birgir Hrafnkelsson, Stefan Siegert, Raphaël Huser, Haakon Bakka, and Árni V. Jóhannesson. Max-and-Smooth: A Two-Step Approach for Approximate Bayesian Inference in Latent Gaussian Models. *Bayesian Analysis*, 16(2):611 – 638, 2021. doi: 10.1214/20-BA1219. URL https://doi.org/10.1214/20-BA1219.

M I Jordan. Serial order: a parallel distributed processing approach. *Technical report*, 5 1986. URL https://www.osti.gov/biblio/6910294.

Michael Jordan, Zoubin Ghahramani, Tommi Jaakkola, and Lawrence Saul. An introduction to variational methods for graphical models. *Machine Learning*, 37:183–233, 01 1999. doi: 10.1023/A:1007665907178.

Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME–Journal of Basic Engineering*, 82(Series D):35–45, 1960.

N. Kantas, A. Doucet, S.S. Singh, and J.M. Maciejowski. An overview of sequential monte carlo methods for parameter estimation in general state-space models. *IFAC Proceedings Volumes*, 42(10):774–785, 2009. ISSN 1474-6670. doi: https://doi.org/10.3182/20090706-3-FR-2004.00129. URL https://www.sciencedirect.com/science/article/pii/S1474667016387432. 15th IFAC Symposium on System Identification.

Nikolas Kantas, Arnaud Doucet, Sumeetpal S. Singh, Jan Maciejowski, and Nicolas Chopin. On Particle Methods for Parameter Estimation in State-Space Models. *Statistical Science*, 30(3):328 – 351, 2015. doi: 10.1214/14-STS511. URL https://doi.org/10.1214/14-STS511.

Peter Karkus, David Hsu, and Wee Sun Lee. Particle filter networks with application to visual localization. In Aude Billard, Anca Dragan, Jan Peters, and Jun Morimoto, editors, *Proceedings of The 2nd Conference on Robot Learning*, volume 87 of *Proceedings of Machine Learning Research*, pages 169–178. PMLR, 29–31 Oct 2018. URL https://proceedings.mlr.press/v87/karkus18a.html.

Hans Kersting and Philipp Hennig. Active uncertainty calibration in Bayesian ODE solvers. In *32nd conference on uncertainty in artificial intelligence (UAI 2016)*, pages 309–318, 2016.

Hans Kersting, Nicholas Krämer, Martin Schiegg, Christian Daniel, Michael Tiemann, and Philipp Hennig. Differentiable likelihoods for fast inversion of likelihood-free dynamical systems. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 5198–5208. PMLR, 13–18 Jul 2020a. URL https://proceedings.mlr.press/v119/kersting20a.html.

Hans Kersting, T. Sullivan, and Philipp Hennig. Convergence rates of Gaussian ODE filters. *Statistics and Computing*, 30:1–26, 11 2020b. doi: 10.1007/s11222-020-09972-4.

Patrick Kidger. *On Neural Differential Equations*. PhD thesis, University of Oxford, 2021. URL https://arxiv.org/pdf/2202.02435.pdf.

Patrick Kidger and Cristian Garcia. Equinox: neural networks in JAX via callable PyTrees and filtered transformations. *Differentiable Programming workshop at Neural Information Processing Systems 2021*, 2021.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017. URL https://arxiv.org/abs/1412.6980.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes, 2022.

Genshiro Kitagawa. Monte carlo filter and smoother for non-gaussian nonlinear state space models. *Journal of Computational and Graphical Statistics*, 5(1):1–25, 1996. doi: 10.1080/10618600.1996.10474692. URL https://www.tandfonline.com/doi/abs/10.1080/10618600.1996.10474692.

Robert C. Kohberger, Donald Scavia, and John W. Wilkinson. A method for parameter sensitivity analysis in differential equation models. *Water Resources Research*, 14(1): 25–29, 1978. doi: https://doi.org/10.1029/WR014i001p00025. URL https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/WR014i001p00025.

S. C. Kou, Benjamin P. Olding, Martin Lysy, and Jun S. Liu. A multiresolution method for parameter estimation of diffusion processes. *Journal of the American Statistical Association*, 107(500):1558–1574, 2012. ISSN 01621459. URL http://www.jstor.org/stable/23427355.

Nicholas Krämer, Nathanael Bosch, Jonathan Schmidt, and Philipp Hennig. Probabilistic ODE solutions in millions of dimensions. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 11634–11649. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/kramer22b.html.

Peter Nicholas Krämer. *Implementing probabilistic numerical solvers for differential equations*. PhD thesis, Universität Tübingen, 2024.

Rahul Krishnan, Uri Shalit, and David Sontag. Structured Inference Networks for Nonlinear State Space Models. *Proceedings of the AAAI Conference on Artificial Intelligence*, 31(1), February 2017. ISSN 2374-3468. doi: 10.1609/aaai.v31i1.10779. URL https://ojs.aaai.org/index.php/AAAI/article/view/10779. Number: 1.

Kasper Kristensen, Anders Nielsen, Casper W. Berg, Hans Skaug, and Bradley M. Bell. Tmb: Automatic differentiation and laplace approximation. *Journal of Statistical Software*, 70(5):1–21, 2016. doi: 10.18637/jss.v070.i05. URL https://www.jstatsoft.org/index.php/jss/article/view/v070i05.

Nicholas Krämer and Philipp Hennig. Stable implementation of probabilistic ode solvers, 2020.

Junpeng Lao and Rémi Louf. Blackjax: A sampling library for JAX, 2020. URL http://github.com/blackjax-devs/blackjax.

Alan Lazarus, Dirk Husmeier, and Theodore Papamarkou. Multiphase MCMC sampling for parameter inference in nonlinear ordinary differential equations. In Amos Storkey and Fernando Perez-Cruz, editors, *Proceedings of the twenty-first international conference on artificial intelligence and statistics*, volume 84 of *Proceedings of machine learning research*, pages 1252–1260. PMLR, 2018. URL https://proceedings.mlr.press/v84/lazarus18a.html.

Edward N Lorenz. Deterministic nonperiodic flow. *Journal of Atmospheric Sciences*, 20: 130–141, 1963. ISSN 1520-0469. URL https://journals.ametsoc.org/doi/pdf/10.1175/1520-0469%281963%29020%3C0130%3ADNF%3E2.0.CO%3B2.

Alfred J. Lotka. Elements of physical biology. *Nature*, 116(2917):461–461, Sep 1925. ISSN 1476-4687. doi: 10.1038/116461b0. URL https://doi.org/10.1038/116461b0.

Martin Lysy, Pranav Subramani, Jonathan Ramkissoon, Mohan Wu, Michelle Ko, and Kanika Chopra. pfjax: Particle filtering methods in jax, 2025. URL https://github.com/mlysy/pfjax.

David J. C. MacKay. A practical Bayesian framework for backpropagation networks. *Neural Computation*, 4(3):448–472, 1992. ISSN 0899-7667, 1530-888X. doi: 10.1162/neco.1992.4.3.448. URL https://direct.mit.edu/neco/article/4/3/448-472/5654.

Gisiro Maruyama. Continuous markov processes and stochastic equations. *Rendiconti del Circolo Matematico di Palermo*, 4(1):48–90, Jan 1955. ISSN 1973-4409. doi: 10.1007/BF02846028. URL https://doi.org/10.1007/BF02846028.

Lawrence M. Murray. Bayesian state-space modelling on high-performance hardware using libbi, 2013. URL https://arxiv.org/abs/1306.3277.

Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer, New York, NY, USA, 2e edition, 2006.

Bernt Øksendal. *Stochastic Differential Equations: An Introduction with Applications*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2003. ISBN 978-3-642-14394-6. doi: 10.1007/978-3-642-14394-6. URL https://doi.org/10.1007/978-3-642-14394-6.

Umberto Picchini and Susanne Ditlevsen. Practical estimation of high dimensional stochastic differential mixed-effects models. *Computational Statistics &; Data Analysis*, 55 (3):1426–1444, March 2011. ISSN 0167-9473. doi: 10.1016/j.csda.2010.10.003. URL http://dx.doi.org/10.1016/j.csda.2010.10.003.

George Poyiadjis, Arnaud Doucet, and Sumeetpal S. Singh. Particle approximations of the score and observed information matrix in state space models with application to parameter estimation. *Biometrika*, 98(1):65–80, 02 2011. ISSN 0006-3444. doi: 10.1093/biomet/asq062. URL https://doi.org/10.1093/biomet/asq062.

Mélanie Prague, Linda Wittkop, Annabelle Collin, Dan Dutartre, Quentin Clairon, Philippe Moireau, Rodolphe Thiébaut, and Boris P. Hejblum. Multi-level modeling of early COVID-19 epidemic dynamics in French regions and estimation of the lockdown impact on infection rate. Preprint, Epidemiology, April 2020.

Christopher Rackauckas, Yingbo Ma, Vaibhav Dixit, Xingjian Guo, Mike Innes, Jarrett Revels, Joakim Nyberg, and Vijay Ivaturi. A comparison of automatic differentiation and continuous sensitivity analysis for derivatives of differential equation solutions. *CoRR*, abs/1812.01892, 2018. URL http://arxiv.org/abs/1812.01892.

H. E. Rauch, F. Tung, and C. T. Striebel. Maximum likelihood estimates of linear dynamic systems. *AIAA Journal*, 3(8):1445–1450, 1965. doi: 10.2514/3.3166. URL https://doi.org/10.2514/3.3166.

Guy Revach, Nir Shlezinger, Xiaoyong Ni, Adrià López Escoriza, Ruud J. G. van Sloun, and Yonina C. Eldar. KalmanNet: Neural Network Aided Kalman Filtering for Partially Known Dynamics. *IEEE Transactions on Signal Processing*, 70:1532–1547, 2022. ISSN 1941-0476. doi: 10.1109/TSP.2022.3158588. URL https://ieeexplore.ieee.org/abstract/document/9733186?casa_token=4QjhW9ZTGFcAAAAA:B2qt4qZtpxXDajAt4wQ6SunOJTXBVBV52lSbQlsYNzfunJUNGpKJd6jhH8d9-djsgtM-YEHe. Conference Name: IEEE Transactions on Signal Processing.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models, 2014.

G. O. Roberts and O. Stramer. On inference for partially observed nonlinear diffusion models using the metropolis-hastings algorithm. *Biometrika*, 88(3):603–621, 2001. ISSN 00063444, 14643510. URL http://www.jstor.org/stable/2673434.

David Ruhe and Patrick Forré. Self-Supervised Inference in State-Space Models, January 2022. URL http://arxiv.org/abs/2107.13349. arXiv:2107.13349 [cs].

D. E. Rumelhart, G. E. Hinton, and R. J. Williams. *Learning internal representations by error propagation*, page 318–362. MIT Press, Cambridge, MA, USA, 1986. ISBN 026268053X.

Tom Ryder. *Variational inference for stochastic processes*. Thesis, Newcastle University, 2021. URL http://theses.ncl.ac.uk/jspui/handle/10443/5521. Accepted: 2022-07-29T13:24:12Z.

Tom Ryder, Andrew Golightly, A. Stephen McGough, and Dennis Prangle. Black-box variational inference for stochastic differential equations. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 4423–4432. PMLR, 10–15 Jul 2018. URL https://proceedings.mlr.press/v80/ryder18a.html.

S. Särkkä and A. Solin. *Applied Stochastic Differential Equations*. Institute of Mathematical Statistics Textbooks. Cambridge University Press, 2019. ISBN 9781316510087. URL https://books.google.ca/books?id=g5SODwAAQBAJ.

Jonathan Schmidt, Nicholas Krämer, and Philipp Hennig. A probabilistic state space model for joint inference from differential equations and data. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 12374–12385. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/6734fa703f6633ab896eecbdfad8953a-Paper.pdf.

Michael Schober, David K Duvenaud, and Philipp Hennig. Probabilistic ODE solvers with Runge-Kutta means. In *Advances in neural information processing systems*, pages 739–747, 2014.

Michael Schober, Simo Särkkä, and Philipp Hennig. A probabilistic model for the numerical solution of initial value problems. *Statistics and Computing*, 29(1):99–122, January 2019. ISSN 0960-3174, 1573-1375. doi: 10.1007/s11222-017-9798-7. URL http://link.springer.com/10.1007/s11222-017-9798-7.

F. Schweppe. Evaluation of likelihood functions for gaussian signals. *IEEE Transactions on Information Theory*, 11(1):61–70, 1965. doi: 10.1109/TIT.1965.1053737.

William Erik Sherwood. *FitzHugh–Nagumo Model*, pages 1–11. Springer New York, New York, NY, 2013. ISBN 978-1-4614-7320-6. doi: 10.1007/978-1-4614-7320-6_147-1. URL https://doi.org/10.1007/978-1-4614-7320-6_147-1.

John Skilling. Bayesian solution of ordinary differential equations. In C. Ray Smith, Gary J. Erickson, and Paul O. Neudorfer, editors, *Maximum Entropy and Bayesian Methods: Seattle, 1991*, Fundamental Theories of Physics, pages 23–37. Springer Netherlands, Dordrecht, 1992. ISBN 978-94-017-2219-3. doi: 10.1007/978-94-017-2219-3_2. URL https://doi.org/10.1007/978-94-017-2219-3_2.

Stan Development Team. *Stan Modeling Language Users Guide and Reference Manual*, 2023. URL https://mc-stan.org.

Ilya Sutskever. *Training Recurrent Neural Networks.* PhD thesis, University of Toronto, Canada, 2013.

Simo Särkkä. *Bayesian Filtering and Smoothing*. Institute of Mathematical Statistics Textbooks. Cambridge University Press, 2013.

Kevin Tan, Giles Hooker, and Edward L. Ionides. Accelerated inference for partially observed markov processes using automatic differentiation, 2024. URL https://arxiv.org/abs/2407.03085.

Onur Teymur, Kostas Zygalakis, and Ben Calderhead. Probabilistic linear multistep methods. In *Advances in Neural Information Processing Systems 29*, pages 4321–4328. Curran Associates, Inc., 2016.

Huaiyu Tian, Yonghong Liu, Yidan Li, Chieh-Hsi Wu, Bin Chen, Moritz U. G. Kraemer, Bingying Li, Jun Cai, Bo Xu, Qiqi Yang, Ben Wang, Peng Yang, Yujun Cui, Yimeng Song, Pai Zheng, Quanyi Wang, Ottar N. Bjornstad, Ruifu Yang, Bryan T. Grenfell, Oliver G. Pybus, and Christopher Dye. An investigation of transmission control measures during the first 50 days of the covid-19 epidemic in china. *Science*, 368(6491):638–642, 2020. doi: 10.1126/science.abb6105. URL https://www.science.org/doi/abs/10.1126/science.abb6105.

Michalis Titsias and Miguel Lázaro-Gredilla. Doubly stochastic variational bayes for non-conjugate inference. In Eric P. Xing and Tony Jebara, editors, *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine*

Learning Research, pages 1971–1979, Bejing, China, 22–24 Jun 2014. PMLR. URL https://proceedings.mlr.press/v32/titsias14.html.

Filip Tronarp, Hans Kersting, Simo Särkkä, and Philipp Hennig. Probabilistic solutions to ordinary differential equations as non-linear Bayesian filtering: a new perspective. *arXiv:1810.03440 [stat]*, October 2018. URL http://arxiv.org/abs/1810.03440. arXiv: 1810.03440.

Filip Tronarp, Simo Särkkä, and Philipp Hennig. Bayesian ODE solvers: The maximum a posteriori estimate. *Statistics and Computing*, 31(3):23, May 2021. ISSN 0960-3174, 1573-1375. doi: 10.1007/s11222-021-09993-7.

Filip Tronarp, Nathanael Bosch, and Philipp Hennig. Fenrir: Physics-enhanced regression for initial value problems. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 21776–21794. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/tronarp22a.html.

Richard Eric Turner and Maneesh Sahani. *Two problems with variational expectation maximisation for time series models*, page 104–124. Cambridge University Press, 2011.

Belinda Tzen and Maxim Raginsky. Neural Stochastic Differential Equations: Deep Latent Gaussian Models in the Diffusion Limit, October 2019. URL http://arxiv.org/abs/1905.09883. arXiv:1905.09883 [cs, stat].

Balth. van der Pol. On "relaxation-oscillations". *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):978–992, 1926. doi: 10.1080/14786442608564127. URL https://doi.org/10.1080/14786442608564127.

Jonathan Wenger, Nicholas Krämer, Marvin Pförtner, Jonathan Schmidt, Nathanael Bosch, Nina Effenberger, Johannes Zenn, Alexandra Gessner, Toni Karvonen, François-Xavier Briol, Maren Mahsereci, and Philipp Hennig. Probnum: Probabilistic numerics in Python, 2021. URL https://arxiv.org/abs/2112.02100.

Philippe Wenk, Alkis Gotovos, Stefan Bauer, Nico S. Gorbach, Andreas Krause, and Joachim M. Buhmann. Fast Gaussian process based gradient matching for parameter identification in systems of nonlinear ODEs. In Kamalika Chaudhuri and Masashi Sugiyama, editors, *Proceedings of the twenty-second international conference on artificial intelligence and statistics*, volume 89 of *Proceedings of machine learning research*, pages 1351–1360. PMLR, 2019. URL https://proceedings.mlr.press/v89/wenk19a.html.

Gavin A. Whitaker, Andrew Golightly, Richard J. Boys, and Chris Sherlock. Bayesian Inference for Diffusion-Driven Mixed-Effects Models. *Bayesian Analysis*, 12(2):435 – 463, 2017. doi: 10.1214/16-BA1009. URL https://doi.org/10.1214/16-BA1009.

Mohan Wu and Martin Lysy. Data-adaptive probabilistic likelihood approximation for ordinary differential equations. In Sanjoy Dasgupta, Stephan Mandt, and Yingzhen Li, editors, *Proceedings of The 27th International Conference on Artificial Intelligence and Statistics*, volume 238 of *Proceedings of Machine Learning Research*, pages 1018–1026. PMLR, 02–04 May 2024. URL https://proceedings.mlr.press/v238/wu24b.html.

Mohan Wu and Martin Lysy. rodeo: Probabilistic methods of parameter inference for odes, 2025a. URL https://github.com/mlysy/rodeo.

Mohan Wu and Martin Lysy. rodeo: Probabilistic methods of parameter inference for ordinary differential equations, 2025b. URL https://arxiv.org/abs/2506.21776.

Shihao Yang, Samuel W. K. Wong, and S. C. Kou. Inference of dynamic systems from noisy and sparse data via manifold-constrained Gaussian processes. *Proceedings of the National Academy of Sciences*, 118(15):e2020397118, 2021. ISSN 0027-8424, 1091-6490. doi: 10.1073/pnas.2020397118. URL http://www.pnas.org/lookup/doi/10.1073/pnas.2020397118.

APPENDICES

Appendix A

Additional Material for Chapter 2

A.1 Proofs for Lemmas and Theorems

We first show that the mean of the RTS smoother converges to the true ODE solution. Since the RTS smoothed mean (2.18) involves both the predicted and updated mean, we need to first establish the convergence of the predicted mean. For that purpose, we borrow the following lemmas from (Kersting et al., 2020b).

Lemma A.1. *Under Assumptions 2.2 and for all sufficiently small step size* $\Delta t > 0$*,*

$$\max_{1 \leq n \leq N} |\boldsymbol{\beta}_n^{(i)}| = O((\Delta t)^{1-i}) \tag{A.1}$$

where $\boldsymbol{\beta}_n$ *is the Kalman gain* (2.16) *for* $i = 0, 1$*.*

Lemma A.2. *Under Assumptions 2.1 and 2.3 and for all sufficiently small step size* $\Delta t > 0$*,*

$$\max_{1 \leq n \leq N} |\boldsymbol{r}_n| = O(\Delta t) \tag{A.2}$$

where $\boldsymbol{r}_n$ *is the Kalman innovation* (2.17)*.*

Equipped with these lemmas, we are ready to prove the following.

Lemma A.3. *Under Assumptions 2.1 to 2.3 and for all sufficiently small step size* $\Delta t > 0$*,*

$$\max_{1 \leq n \leq N} |\boldsymbol{\mu}_{n|n}^{(i)} - \boldsymbol{\mu}_{n|n-1}^{(i)}| = O((\Delta t)^{2-i}) \tag{A.3}$$

where $\boldsymbol{\mu}_{n|n}$ *is the Kalman updated mean and* $\boldsymbol{\mu}_{n|n-1}$ *is the Kalman predicted mean.*

Proof. A direct substitution of the Kalman update equation (2.14) with Lemma A.1 and A.2 gives

$$\begin{aligned}\max_{1\le n\le N} |\boldsymbol{\mu}_{n|n}^{(i)} - \boldsymbol{\mu}_{n|n-1}^{(i)}| &= \max_{1\le n\le N} |\boldsymbol{\beta}_n^{(i)} \boldsymbol{r}_n| \\ &= O((\Delta t)^{2-i}).\end{aligned}$$

□

The only quantity in the smoothing mean equation (2.18) remaining with unknown convergence is the matrix $\boldsymbol{A}_n$, which involve both the predicted and updated variances. Therefore, we need to borrow the results of the following lemmas (Kersting et al., 2020b).

Lemma A.4. *Under Assumptions 2.1 to 2.3, for all sufficiently small step size $\Delta t > 0$ and at least one of i, j is not equal to 0,*

$$\max_{0\le n\le N} |\boldsymbol{\Sigma}_{n|n}^{(i,j)}| = 0. \tag{A.4}$$

Remarkably, Lemma A.4 states that all values, except for the $0, 0$-th element in the updated covariance matrix, are 0. This is due to Assumption 2.3, where we assume measurements of the first derivative have zero variance. This lemma, together with Theorem 2.2, establishes the order of convergence for the entire updated variance. Similarly, we have the following results for the predicted variance.

Lemma A.5. *Under Assumptions 2.1 to 2.3 and for all sufficiently small step size $\Delta t > 0$,*

$$\begin{aligned}\max_{1\le n\le N} \boldsymbol{\Sigma}_{n|n-1}^{(0,0)} &= O((\Delta t)^2) \\ \max_{1\le n\le N} |\boldsymbol{\Sigma}_{n|n-1}^{(1,0)}| &= O((\Delta t)^2) \\ \max_{1\le n\le N} \boldsymbol{\Sigma}_{n|n-1}^{(1,1)} &= O(\Delta t)\end{aligned} \tag{A.5}$$

where $\boldsymbol{\Sigma}_{n|n-1}$ is the Kalman predicted variance.

Lemma A.5 is a direct result of applying the Kalman predict equations to Lemma A.4 and Theorem 2.2, assuming the IBM prior (2.7). With the help of these lemmas and Theorem 2.2, we can establish the following orders of convergence for the smoothing matrix (2.19).

Lemma A.6. *Under Assumptions 2.1 to 2.3, for all sufficiently small step size* $\Delta t > 0$*,*

$$\begin{aligned}
\max_{1\le n\le N-1} |\boldsymbol{A}_n^{(0,0)}| &= O(1) \\
\max_{1\le n\le N-1} |\boldsymbol{A}_n^{(0,1)}| &= O(\Delta t) \\
\max_{1\le n\le N-1} |\boldsymbol{A}_n^{(1,0)}| &= 0 \\
\max_{1\le n\le N-1} |\boldsymbol{A}_n^{(1,1)}| &= 0
\end{aligned} \tag{A.6}$$

where $\boldsymbol{A}_n$ *is the smoothing matrix in* (2.19)*.*

Proof. First we note that the IBM prior (2.7) gives the convergence rate for $\boldsymbol{Q} = \begin{bmatrix} 1 & O(\Delta t) \\ 0 & 1 \end{bmatrix}$. Using Theorem 2.2, Lemmas A.4 and A.5 we have

$$\begin{aligned}
\boldsymbol{A}_n &= \boldsymbol{\Sigma}_{n|n}\boldsymbol{Q}^\top \boldsymbol{\Sigma}_{n+1|n}^{-1} \\
&= \begin{bmatrix} O((\Delta t)^2) & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ O(\Delta t) & 1 \end{bmatrix} \begin{bmatrix} O((\Delta t)^2) & O((\Delta t)^2) \\ O((\Delta t)^2) & O(\Delta t) \end{bmatrix}^{-1} \\
&= \begin{bmatrix} O((\Delta t)^2) & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ O(\Delta t) & 1 \end{bmatrix} \begin{bmatrix} O((1/\Delta t)^2) & O(1/\Delta t) \\ O(1/\Delta t) & O(1/\Delta t) \end{bmatrix} \\
&= \begin{bmatrix} O(1) & O(\Delta t) \\ 0 & 0 \end{bmatrix}
\end{aligned}$$

where we used the standard formula to find the inverse of a 2×2 matrix on line 3 and noting that $O((\Delta t)^3 + (\Delta t)^4) = O((\Delta t)^3)$ in the determinant. □

With Lemma A.6, we can now compare the difference between the RTS smoothed mean and the Kalman updated mean, leading us to the following lemma.

Lemma A.7. *Under Assumptions 2.1 to 2.3 and for all sufficiently small step size* $\Delta t > 0$*,*

$$\begin{aligned}
\max_{0\le n\le N} |\boldsymbol{\mu}_{n|N}^{(0)} - \boldsymbol{\mu}_{n|n}^{(0)}| &= O(\Delta t) \\
\max_{0\le n\le N} |\boldsymbol{\mu}_{n|N}^{(1)} - \boldsymbol{\mu}_{n|n}^{(1)}| &= 0
\end{aligned} \tag{A.7}$$

where $\boldsymbol{\mu}_{n|N}$ *is the RTS smoothed mean and* $\boldsymbol{\mu}_{n|n}$ *is the Kalman updated mean.*

Proof. Recall that the RTS smoother operates backward in time, starting at time point t_N. We will first prove the following statement using induction

$$\text{abs}(\boldsymbol{\mu}_{N-i|N} - \boldsymbol{\mu}_{N-i|N-i}) = \begin{bmatrix} iO((\Delta t)^2)) \\ 0 \end{bmatrix},$$

where $\text{abs}(\boldsymbol{\mu})$ denotes element-wise absolute value. Here, we keep track of the constant i due to the fact that this constant depends on N and $\Delta t = T/N$. For $i = 0$, we have the updated mean equal to the smoothed mean, which trivially satisfies the induction statement. For the induction hypothesis, we assume this holds for $i = k$ and prove that it holds for $i = k + 1$. By the RTS smoother equation (2.18),

$$\begin{aligned}
&\text{abs}(\boldsymbol{\mu}_{N-k-1|N} - \boldsymbol{\mu}_{N-k-1|N-k-1}) \\
=&\,\text{abs}(\boldsymbol{A}_{N-k-1}(\boldsymbol{\mu}_{N-k|N} - \boldsymbol{\mu}_{N-k|N-k-1})) \\
\leq&\,\text{abs}(\boldsymbol{A}_{N-k-1}(\boldsymbol{\mu}_{N-k|N} - \boldsymbol{\mu}_{N-k|N-k})) + \text{abs}(\boldsymbol{A}_{N-k-1}(\boldsymbol{\mu}_{N-k|N-k} - \boldsymbol{\mu}_{N-k|N-k-1})) \\
=&\begin{bmatrix} O(1) & O(\Delta t) \\ 0 & 0 \end{bmatrix} \left(\begin{bmatrix} kO((\Delta t)^2) \\ 0 \end{bmatrix} + \begin{bmatrix} O((\Delta t)^2) \\ O(\Delta t) \end{bmatrix} \right) \\
=&\begin{bmatrix} (k+1)O((\Delta t)^2)) \\ 0 \end{bmatrix},
\end{aligned}$$

where we used the triangle inequality on line 3 and Lemma A.3, Lemma A.6 and the induction hypothesis on line 4. Thus, this holds true for $i = k + 1$, establishing the inductive step. Note that $0 \leq i \leq N$ implies that $0 \leq n \leq N$ as $n = N - i$, and therefore we have

$$\begin{aligned}
\max_{0 \leq n \leq N} |\boldsymbol{\mu}^{(0)}_{n|N} - \boldsymbol{\mu}^{(0)}_{n|n}| &\leq NO((\Delta t)^2) \\
&= O(\Delta t).
\end{aligned}$$

A direct application of the above result gives

$$\max_{0 \leq n \leq N} |\boldsymbol{\mu}^{(1)}_{n|N} - \boldsymbol{\mu}^{(1)}_{n|n}| = 0.$$

□

Lemma A.7 provides the bound on how far apart the RTS smoothed mean is compared to the Kalman updated mean. Notably, the first derivative of both means are exactly equal, which is due to Assumption 2.3. This lemma allows for a short-and-sweet proof of the Theorem 2.3.

Proof. A direct application of Lemma A.7, Theorem 2.1 and the triangle inequality gives

$$\begin{aligned}\max_{0\le n\le N} |\boldsymbol{\mu}^{(0)}_{n|N} - x_n| &\le \max_{0\le n\le N} |\boldsymbol{\mu}^{(0)}_{n|N} - \boldsymbol{\mu}^{(0)}_{n|n}| + |\boldsymbol{\mu}^{(0)}_{n|n} - x_n| \\ &= O(\Delta t) + O(\Delta t) \\ &= O(\Delta t).\end{aligned}$$

□

Next, we prove the same level of convergence for the RTS smoother variance. Remarkably, we have the following stronger result due to our choice of the IBM prior in Section 2.2.1. The following is a proof for Theorem 2.4.

Proof. We prove this using induction starting at t_N. For the base case $i = 0$, this is clearly satisfied since the smoothed variance is equal to the updated variance at t_N. Suppose, for the inductive hypothesis, this holds for $i = n$, such that $\boldsymbol{\Sigma}_{N-n|N} = \boldsymbol{\Sigma}_{N-n|N-n}$. We prove this holds true for $i = n + 1$. For ease of notation, let $k = N - n$, then the smoother equation (2.18) gives

$$\begin{aligned}\boldsymbol{\Sigma}_{k-1|N} - \boldsymbol{\Sigma}_{k-1|k-1} &= \boldsymbol{A}_{k-1}(\boldsymbol{\Sigma}_{k|N} - \boldsymbol{\Sigma}_{k|k-1})\boldsymbol{A}^\top_{k-1} \\ &= \boldsymbol{A}_{k-1}(\boldsymbol{\Sigma}_{k|k} - \boldsymbol{\Sigma}_{k|k-1})\boldsymbol{A}^\top_{k-1}\end{aligned}$$

where we used the inductive hypothesis $\boldsymbol{\Sigma}_{k|N} = \boldsymbol{\Sigma}_{k|k}$. Substituting the Kalman update equation (2.14) to the right-hand side of the above equation gives

$$\begin{aligned}&\boldsymbol{A}_{k-1}(\boldsymbol{\Sigma}_{k|k} - \boldsymbol{\Sigma}_{k|k-1})\boldsymbol{A}^\top_{k-1} \\ =&\boldsymbol{\Sigma}_{k-1|k-1}\boldsymbol{Q}^\top\boldsymbol{\Sigma}^{-1}_{k|k-1}(\boldsymbol{\beta}_k\boldsymbol{W}\boldsymbol{\Sigma}_{k|k-1})\boldsymbol{\Sigma}^{-1}_{k|k-1}\boldsymbol{Q}\boldsymbol{\Sigma}_{k-1|k-1} \\ =&\boldsymbol{\Sigma}_{k-1|k-1}\boldsymbol{Q}^\top\boldsymbol{W}^\top(\boldsymbol{W}\boldsymbol{\Sigma}_{k|k-1}\boldsymbol{W}^\top)^{-1}\boldsymbol{W}\boldsymbol{Q}\boldsymbol{\Sigma}_{k-1|k-1},\end{aligned} \tag{A.8}$$

where we used Assumption 2.3 and the Kalman gain equation (2.16). Note that the first three terms of (A.8) are exactly the transpose of the last three terms. So, consider the first three terms, then by Theorem 2.2 and Lemma A.4 we have

$$\begin{aligned}\boldsymbol{\Sigma}_{k-1|k-1}\boldsymbol{Q}^\top\boldsymbol{W}^\top &= \begin{bmatrix} a & 0 \\ 0 & 0 \end{bmatrix}\begin{bmatrix} 1 & 0 \\ b & 1 \end{bmatrix}\begin{bmatrix} 0 \\ 1 \end{bmatrix} \\ &= \begin{bmatrix} 0 \\ 0 \end{bmatrix}\end{aligned}$$

for some $a, b > 0$ which means that $\boldsymbol{\Sigma}_{k-1|N} - \boldsymbol{\Sigma}_{k-1|k-1} = \mathbf{0}$. Thus, we have proven the inductive step implying $\boldsymbol{\Sigma}_{n|N} = \boldsymbol{\Sigma}_{n|n}$ for all $0 \le n \le N$. □

A.2 Kalman recursions

The Kalman recursions used in Algorithms 2.1, 2.4, 2.2, 3.1 and B.1 are formulated in terms of the general Gaussian state space model

$$\begin{aligned} \boldsymbol{X}_n &= \boldsymbol{Q}_n \boldsymbol{X}_{n-1} + \boldsymbol{c}_n + \boldsymbol{R}_n^{1/2} \boldsymbol{\epsilon}_n \\ \boldsymbol{Z}_n &= \boldsymbol{W}_n \boldsymbol{X}_n + \boldsymbol{a}_n + \boldsymbol{V}_n^{1/2} \boldsymbol{\eta}_n \end{aligned}, \qquad \boldsymbol{\epsilon}_n, \boldsymbol{\eta}_n \overset{\text{ind}}{\sim} \text{Normal}(\boldsymbol{0}, \boldsymbol{I}). \tag{A.9}$$

Throughout, we use the notation $\boldsymbol{\mu}_{n|m} = E[\boldsymbol{X}_n \mid \boldsymbol{Z}_{0:m}]$ and $\boldsymbol{\Sigma}_{n|m} = \text{var}(\boldsymbol{X}_n \mid \boldsymbol{Z}_{0:m})$.

Algorithm A.1 Standard Kalman filtering and smoothing recursions.

procedure `kalman_predict`$(\boldsymbol{\mu}_{n-1|n-1}, \boldsymbol{\Sigma}_{n-1|n-1}, \boldsymbol{c}_n, \boldsymbol{Q}_n, \boldsymbol{R}_n)$
  $\boldsymbol{\mu}_{n|n-1} \leftarrow \boldsymbol{Q}_n \boldsymbol{\mu}_{n-1|n-1} + \boldsymbol{c}_n$
  $\boldsymbol{\Sigma}_{n|n-1} \leftarrow \boldsymbol{Q}_n \boldsymbol{\Sigma}_{n-1|n-1} \boldsymbol{Q}_n^\top + \boldsymbol{R}_n$
  return $\boldsymbol{\mu}_{n|n-1}, \boldsymbol{\Sigma}_{n|n-1}$

procedure `kalman_update`$(\boldsymbol{\mu}_{n|n-1}, \boldsymbol{\Sigma}_{n|n-1}, \boldsymbol{Z}_n, \boldsymbol{a}_n, \boldsymbol{W}_n, \boldsymbol{V}_n)$
  $\boldsymbol{A}_n \leftarrow \boldsymbol{\Sigma}_{n|n-1} \boldsymbol{W}_n^\top [\boldsymbol{W}_n \boldsymbol{\Sigma}_{n|n-1} \boldsymbol{W}_n^\top + \boldsymbol{V}_n]^{-1}$
  $\boldsymbol{\mu}_{n|n} \leftarrow \boldsymbol{\mu}_{n|n-1} + \boldsymbol{A}_n (\boldsymbol{Z}_n - \boldsymbol{W}_n \boldsymbol{\mu}_{n|n-1} - \boldsymbol{a}_n)$
  $\boldsymbol{\Sigma}_{n|n} \leftarrow \boldsymbol{\Sigma}_{n|n-1} - \boldsymbol{A}_n \boldsymbol{W}_n \boldsymbol{\Sigma}_{n|n-1}$
  return $\boldsymbol{\mu}_{n|n}, \boldsymbol{\Sigma}_{n|n}$

procedure `kalman_smooth`$(\boldsymbol{\mu}_{n+1|N}, \boldsymbol{\Sigma}_{n+1|N}, \boldsymbol{\mu}_{n|n}, \boldsymbol{\Sigma}_{n|n}, \boldsymbol{\mu}_{n+1|n}, \boldsymbol{\Sigma}_{n+1|n}, \boldsymbol{Q}_{n+1})$
  $\boldsymbol{A}_n \leftarrow \boldsymbol{\Sigma}_{n|n} \boldsymbol{Q}_{n+1}^\top \boldsymbol{\Sigma}_{n+1|n}^{-1}$
  $\boldsymbol{\mu}_{n|N} \leftarrow \boldsymbol{\mu}_{n|n} + \boldsymbol{A}_n (\boldsymbol{\mu}_{n+1|N} - \boldsymbol{\mu}_{n+1|n})$
  $\boldsymbol{\Sigma}_{n|N} \leftarrow \boldsymbol{\Sigma}_{n|n} + \boldsymbol{A}_n (\boldsymbol{\Sigma}_{n+1|N} - \boldsymbol{\Sigma}_{n+1|n}) \boldsymbol{A}_n^\top$
  return $\boldsymbol{\mu}_{n|N}, \boldsymbol{\Sigma}_{n|N}$

procedure `kalman_sample`$(\boldsymbol{x}_{n+1}, \boldsymbol{\mu}_{n|n}, \boldsymbol{\Sigma}_{n|n}, \boldsymbol{\mu}_{n+1|n}, \boldsymbol{\Sigma}_{n+1|n}, \boldsymbol{Q}_{n+1})$
  $\boldsymbol{A}_n \leftarrow \boldsymbol{\Sigma}_{n|n} \boldsymbol{Q}_{n+1}^\top \boldsymbol{\Sigma}_{n+1|n}^{-1}$
  $\tilde{\boldsymbol{\mu}}_{n|N} \leftarrow \boldsymbol{\mu}_{n|n} + \boldsymbol{A}_n (\boldsymbol{x}_{n+1} - \boldsymbol{\mu}_{n+1|n})$ ▷ $\tilde{\boldsymbol{\mu}}_{n|N} = E[\boldsymbol{X}_n \mid \boldsymbol{X}_{n+1:N}, \boldsymbol{Z}_{0:N}]$
  $\tilde{\boldsymbol{\Sigma}}_{n|N} \leftarrow \boldsymbol{\Sigma}_{n|n} - \boldsymbol{A}_n \boldsymbol{Q}_{n+1} \boldsymbol{\Sigma}_{n|n}$ ▷ $\tilde{\boldsymbol{\Sigma}}_{n|N} = \mathrm{var}[\boldsymbol{X}_n \mid \boldsymbol{X}_{n+1:N}, \boldsymbol{Z}_{0:N}]$
  return $\tilde{\boldsymbol{\mu}}_{n|N}, \tilde{\boldsymbol{\Sigma}}_{n|N}$

procedure `kalman_forecast`$(\boldsymbol{\mu}_{n|n-1}, \boldsymbol{\Sigma}_{n|n-1}, \boldsymbol{a}_n, \boldsymbol{W}_n, \boldsymbol{V}_n)$
  $\boldsymbol{\lambda}_n \leftarrow \boldsymbol{W}_n \boldsymbol{\mu}_{n|n-1} + \boldsymbol{a}_n$ ▷ $\boldsymbol{\lambda}_{n|n-1} = E[\boldsymbol{Z}_n \mid \boldsymbol{Z}_{0:n-1}]$
  $\boldsymbol{\Omega}_{n|n-1} \leftarrow \boldsymbol{W}_n \boldsymbol{\mu}_{n|n-1} \boldsymbol{W}_n^\top + \boldsymbol{V}_n$ ▷ $\boldsymbol{\Omega}_{n|n-1} = \mathrm{var}[\boldsymbol{Z}_n \mid \boldsymbol{Z}_{0:n-1}]$
  return $\boldsymbol{\lambda}_{n|n-1}, \boldsymbol{\Omega}_{n|n-1}$

procedure `kalman_cond`$(\boldsymbol{\mu}_{n|n}, \boldsymbol{\Sigma}_{n|n}, \boldsymbol{\mu}_{n+1|n}, \boldsymbol{\Sigma}_{n+1|n}, \boldsymbol{Q}_{n+1})$
  $\boldsymbol{A}_n \leftarrow \boldsymbol{\Sigma}_{n|n} \boldsymbol{Q}_{n+1}^\top \boldsymbol{\Sigma}_{n+1|n}^{-1}$
  $\boldsymbol{b}_n \leftarrow \boldsymbol{\mu}_{n|n} - \boldsymbol{A}_n \boldsymbol{\mu}_{n+1|n}$
  $\boldsymbol{C}_n \leftarrow \boldsymbol{\Sigma}_{n|n} - \boldsymbol{A}_n \boldsymbol{Q}_n \boldsymbol{\Sigma}_{n|n}$
  return $\boldsymbol{A}_n, \boldsymbol{b}_n, \boldsymbol{C}_n$ ▷ $\boldsymbol{X}_n \mid \boldsymbol{X}_{n+1} \sim \mathrm{Normal}(\boldsymbol{A}_n \boldsymbol{X}_{n+1} + \boldsymbol{b}_n, \boldsymbol{C}_n)$

Appendix B

Additional Material for Chapter 3

B.1 DALTON for Non-Gaussian Observations

Recall in Section 3.2.2 that we denote $\boldsymbol{s}_{n(i)}$ and $\boldsymbol{u}_{n(i)}$ to be the observed and unobserved components of $\boldsymbol{x}(t)$ at time $t = t'_i$. In order to present Algorithm B.1, we introduce the notation $\boldsymbol{s}_{n(i)} = \boldsymbol{D}_i \boldsymbol{X}_{n(i)}$, where $\boldsymbol{D}_i$ is a mask matrix of zeros and ones, so as to more naturally relate to the Gaussian version of DALTON in Algorithm 3.1.

Algorithm B.1 DALTON probabilistic ODE likelihood approximation for non-Gaussian observations.

1: **procedure** `daltonng`$(\boldsymbol{f}(\boldsymbol{x}, t, \boldsymbol{\theta}), \boldsymbol{v}_{\boldsymbol{\theta}}, \boldsymbol{Q}_{\boldsymbol{\eta}}, \boldsymbol{R}_{\boldsymbol{\eta}}, \boldsymbol{Y}_{0:M}, \boldsymbol{g}_{0:M}(\boldsymbol{Y}, \boldsymbol{X}, \boldsymbol{\phi}))$

2: $\quad \boldsymbol{W}^{(1)} \leftarrow \begin{bmatrix} 0 & 1 \end{bmatrix}, \qquad \boldsymbol{W}^{(0)} \leftarrow \begin{bmatrix} 1 & 0 \end{bmatrix}$

3: $\quad \boldsymbol{\mu}_{0|0}, \Sigma_{0|0}, \boldsymbol{Z}_{1:N}, \leftarrow \boldsymbol{v}, \boldsymbol{0}, \boldsymbol{0}$ ▷ Initialization

4: $\quad \ell_{xz}, \ell_{xyz}, \ell_y \leftarrow 0, 0, 0$

5: $\quad i \leftarrow 0$ ▷ Used to map t_n to t'_i

6: ▷ Lines 7-22 approximate $\log p(\boldsymbol{X}_{0:N} \mid \hat{\boldsymbol{Y}}_{0:M}, \boldsymbol{Z}_{1:N} = \boldsymbol{0}, \boldsymbol{\Theta})$

7: $\quad$ **for** $n = 1 : N$ **do**

8: $\quad\quad \boldsymbol{\mu}_{n|n-1}, \boldsymbol{\Sigma}_{n|n-1} \leftarrow$ `kalman_predict`$(\boldsymbol{\mu}_{n-1|n-1}, \boldsymbol{\Sigma}_{n-1|n-1}, \boldsymbol{Q}_{\boldsymbol{\eta}}, \boldsymbol{R}_{\boldsymbol{\eta}})$

9: $\quad\quad \boldsymbol{a}_n, \boldsymbol{B}_n, \boldsymbol{V}_n \leftarrow$ `interrogate`$(\boldsymbol{\mu}_{n|n-1}, \boldsymbol{\Sigma}_{n|n-1}, \boldsymbol{f}(\boldsymbol{x}, t, \boldsymbol{\theta}))$

10: $\quad\quad$ **if** $t_n = t_{n(i)}$ **then**

11: $\quad\quad\quad \boldsymbol{g}_{(1)} \leftarrow \frac{\partial}{\partial \boldsymbol{s}_{n(i)}} g_i(\boldsymbol{Y}_i, \boldsymbol{D}_i \boldsymbol{\mu}_{n|n-1}, \boldsymbol{\phi})$

12: $\quad\quad\quad \boldsymbol{g}_{(2)} \leftarrow \frac{\partial^2}{\partial \boldsymbol{s}_{n(i)} \partial \boldsymbol{s}'_{n(i)}} g_i(\boldsymbol{Y}_i, \boldsymbol{D}_i \boldsymbol{\mu}_{n|n-1}, \boldsymbol{\phi})$

13: $\quad\quad\quad \hat{\boldsymbol{Y}}_i \leftarrow \boldsymbol{D}_i \boldsymbol{\mu}_{n|n-1} - \boldsymbol{g}_{(2)}^{-1} \boldsymbol{g}_{(1)}$ ▷ Compute pseudo-observations

14: $\quad\quad\quad \boldsymbol{Z}_n \leftarrow \begin{bmatrix} \boldsymbol{Z}_n \\ \hat{\boldsymbol{Y}}_i \end{bmatrix}, \qquad \boldsymbol{W} \leftarrow \begin{bmatrix} \boldsymbol{W}^{(1)} \\ \boldsymbol{D}_i \end{bmatrix}, \qquad \boldsymbol{B}_n \leftarrow \begin{bmatrix} \boldsymbol{B}_n \boldsymbol{W}^{(0)} \\ \boldsymbol{0} \end{bmatrix}$

15: $\quad\quad\quad \boldsymbol{a}_n \leftarrow \begin{bmatrix} \boldsymbol{a}_n \\ \boldsymbol{0} \end{bmatrix}, \qquad \boldsymbol{V}_n \leftarrow \begin{bmatrix} \boldsymbol{V}_n & \boldsymbol{0} \\ \boldsymbol{0} & -\boldsymbol{g}_{(2)}^{-1} \end{bmatrix}$

16: $\quad\quad\quad i \leftarrow i + 1$

17: $\quad\quad \boldsymbol{\mu}_{n|n}, \boldsymbol{\Sigma}_{n|n} \leftarrow$ `kalman_update`$(\boldsymbol{\mu}_{n|n-1}, \boldsymbol{\Sigma}_{n|n-1}, \boldsymbol{Z}_n, \boldsymbol{a}_n, \boldsymbol{W} + \boldsymbol{B}_n, \boldsymbol{V}_n)$

18: $\quad$ **for** $n = N - 1 : 1$ **do**

19: $\quad\quad \boldsymbol{\mu}_{n|N}, \boldsymbol{\Sigma}_{n|N} \leftarrow$ `kalman_smooth`$(\boldsymbol{\mu}_{n+1|N}, \boldsymbol{\Sigma}_{n+1|N}, \boldsymbol{\mu}_{n|n}, \boldsymbol{\Sigma}_{n|n}, \boldsymbol{\mu}_{n+1|n}, \boldsymbol{\Sigma}_{n+1|n}, \boldsymbol{Q}_{\boldsymbol{\eta}})$

20: $\quad\quad \boldsymbol{\mu}_n, \boldsymbol{\Sigma}_n \leftarrow$ `kalman_sample`$(\boldsymbol{\mu}_{n+1|N}, \boldsymbol{\mu}_{n|n}, \boldsymbol{\Sigma}_{n|n}, \boldsymbol{\mu}_{n+1|n}, \boldsymbol{\Sigma}_{n+1|n}, \boldsymbol{Q}_{\boldsymbol{\eta}})$

21: $\quad\quad \ell_{xyz} \leftarrow \ell_{xyz} +$ `normal_logpdf`$(\boldsymbol{\mu}_{n|N}; \boldsymbol{\mu}_n, \boldsymbol{\Sigma}_n)$

22: $\quad \ell_{xyz} \leftarrow \ell_{xyz} +$ `normal_logpdf`$(\boldsymbol{\mu}_{N|N}; \boldsymbol{\mu}_{N|N}, \boldsymbol{\Sigma}_{N|N})$

Algorithm B.2 Non-Gaussian DALTON (continued)

23: ▷ Lines 24-33 approximate $\log p(\boldsymbol{X}_{0:N} \mid \boldsymbol{Z}_{1:N} = \boldsymbol{0}, \boldsymbol{\Theta})$

24: $\boldsymbol{Z}_{1:N} \leftarrow \boldsymbol{0}$ ▷ Reset $\boldsymbol{Z}_{1:N} = \boldsymbol{0}$

25: $\boldsymbol{\mu}'_{0:N|N} \leftarrow \boldsymbol{\mu}_{0:N|N}$ ▷ Store $\boldsymbol{\mu}_{0:N|N} = E[\boldsymbol{X}_{0:N} \mid \boldsymbol{Z}_{1:N}, \hat{\boldsymbol{Y}}_{0:N}]$ for later

26: **for** $n = 1 : N$ **do**

27: $\boldsymbol{\mu}_{n|n-1}, \boldsymbol{\Sigma}_{n|n-1} \leftarrow \texttt{kalman_predict}(\boldsymbol{\mu}_{n-1|n-1}, \boldsymbol{\Sigma}_{n-1|n-1}, \boldsymbol{Q}_{\boldsymbol{\eta}}, \boldsymbol{R}_{\boldsymbol{\eta}})$

28: $\boldsymbol{a}_n, \boldsymbol{B}_n, \boldsymbol{V}_n \leftarrow \texttt{interrogate}(\boldsymbol{\mu}_{n|n-1}, \boldsymbol{\Sigma}_{n|n-1}, \boldsymbol{f}(\boldsymbol{x}, t, \boldsymbol{\theta}))$

29: $\boldsymbol{\mu}_{n|n}, \boldsymbol{\Sigma}_{n|n} \leftarrow \texttt{kalman_update}(\boldsymbol{\mu}_{n|n-1}, \boldsymbol{\Sigma}_{n|n-1}, \boldsymbol{Z}_n, \boldsymbol{a}_n, \boldsymbol{W}_1 + \boldsymbol{B}_n \boldsymbol{W}_0, \boldsymbol{V}_n)$

30: **for** $n = N - 1 : 1$ **do**

31: $\boldsymbol{\mu}_n, \boldsymbol{\Sigma}_n \leftarrow \texttt{kalman_sample}(\boldsymbol{\mu}'_{n+1|N}, \boldsymbol{\mu}_{n|n}, \boldsymbol{\Sigma}_{n|n}, \boldsymbol{\mu}_{n+1|n}, \boldsymbol{\Sigma}_{n+1|n}, \boldsymbol{Q}_{\boldsymbol{\eta}})$

32: $\ell_{xz} \leftarrow \ell_{xz} + \texttt{normal_logpdf}(\boldsymbol{\mu}'_{n|N}; \boldsymbol{\mu}_n, \boldsymbol{\Sigma}_n)$

33: $\ell_{xz} \leftarrow \ell_{xz} + \texttt{normal_logpdf}(\boldsymbol{\mu}'_{N|N}; \boldsymbol{\mu}_{N|N}, \boldsymbol{\Sigma}_{N|N})$

34: ▷ Lines 35-36 approximate $\log p(\boldsymbol{Y}_{0:M} \mid \boldsymbol{X}_{0:N}, \boldsymbol{\phi})$

35: **for** $i = 0 : M$ **do**

36: $\ell_y \leftarrow \ell_y + g_i(\boldsymbol{Y}_i, \boldsymbol{D}_i \boldsymbol{\mu}'_{n(i)|N}, \boldsymbol{\phi})$

37:

38: **return** $\ell_{xz} - \ell_y - \ell_{xyz}$ ▷ Estimate of $\log p(\boldsymbol{Y}_{0:M} \mid \boldsymbol{Z}_{1:N} = \boldsymbol{0}, \boldsymbol{\Theta})$

B.2 Parameter estimation experiment details

In the experiments of Section 3.3, we take the observation model parameter $\boldsymbol{\phi}$ to be known, such that parameter estimation is only conducted on the parameter $\boldsymbol{\eta}$ to tune the IBM prior in Section 2.2.1 and the ODE model parameters $\boldsymbol{\theta}$. For the former we assume a flat prior and for the latter we assume independent $\text{Normal}(0, 10^2)$ priors on either θ_r or $\log \theta_r$, $r = 1, \ldots, D$, depending on whether θ_r is unbounded or $\theta_r > 0$. We choose these priors largely to allow for parameter priors to be uninformative. As such, the Bayesian normal approximation is

$$\boldsymbol{\theta} \mid \boldsymbol{Y}_{0:M} \approx \text{Normal}(\hat{\boldsymbol{\theta}}, \hat{\boldsymbol{V}}_{\boldsymbol{\theta}}), \tag{B.1}$$

where $(\hat{\boldsymbol{\theta}}, \hat{\boldsymbol{\eta}}) = \arg\max_{(\boldsymbol{\theta}, \boldsymbol{\eta})} \log p(\boldsymbol{\theta}, \boldsymbol{\eta} \mid \boldsymbol{Y}_{0:M})$ is computed using the Newton-CG optimization algorithm (Nocedal and Wright, 2006) as implemented in the **JAXopt** library (Blondel et al., 2021). However, the Hessian is taken only with respect to $\boldsymbol{\theta}$ and not $\boldsymbol{\eta}$, namely, $\hat{\boldsymbol{V}}_{\boldsymbol{\theta}} = -\left[\frac{\partial^2}{\partial \boldsymbol{\theta} \partial \boldsymbol{\theta}'} \log p(\hat{\boldsymbol{\Theta}} \mid \boldsymbol{Y}_{0:M})\right]^{-1}$ which has been previously advocated by Tronarp et al. (2022). We find this produced slightly better results for the posteriors of $\boldsymbol{\theta}$ compared to propagating the uncertainty of $\boldsymbol{\eta}$ in the Hessian calculation.

B.3 Lorenz63 Model Extra Figures

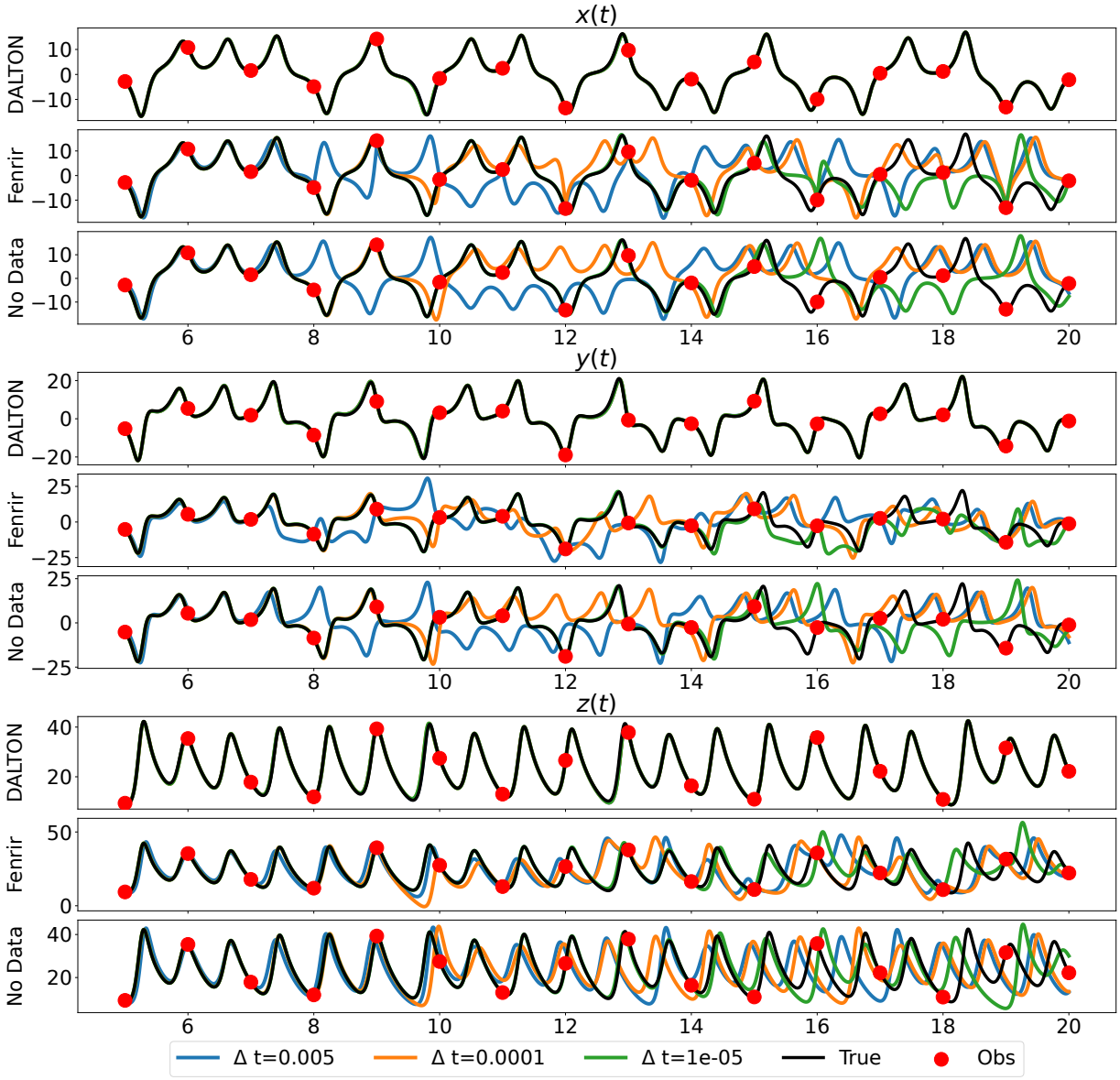

Figure B.1: The full latent variables for the Lorenz63 model calculated using various solvers.

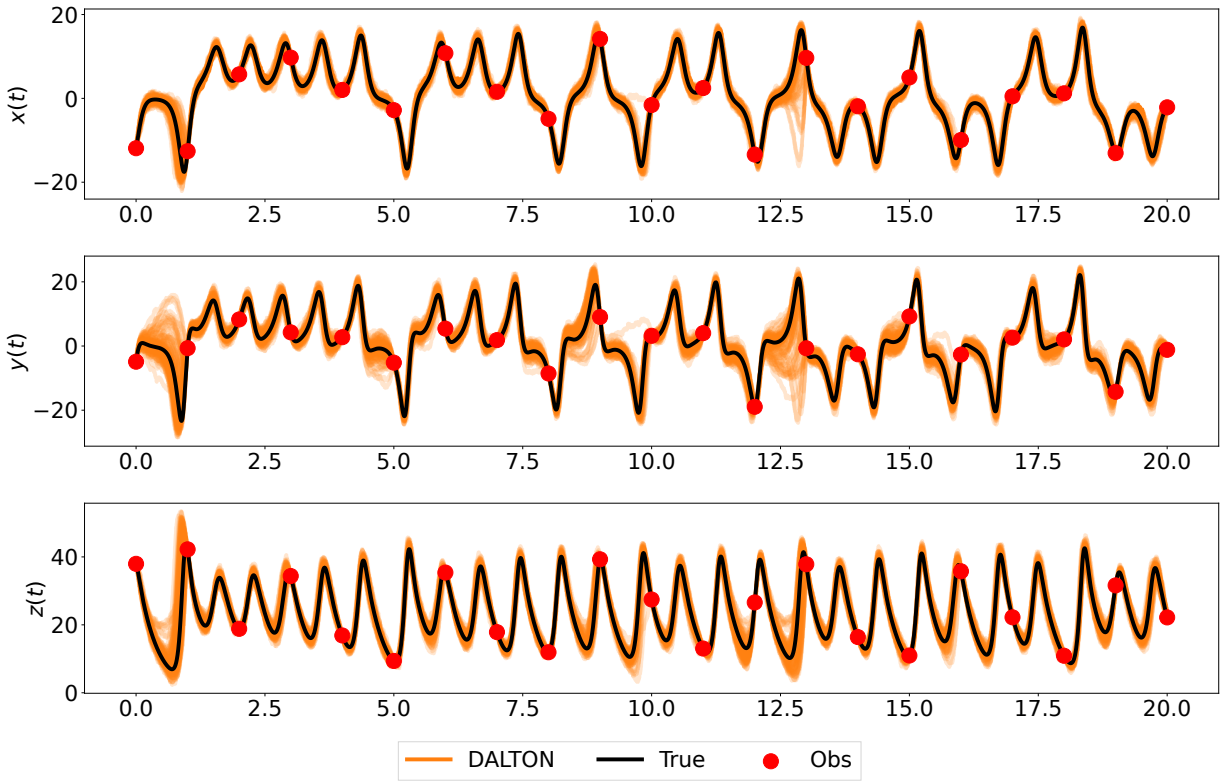

Figure B.2: ODE posteriors for the Lorenz63 model using DALTON. Extended version of Figure 3.5.