

# **Statistical methods for sequential decision making and inference**

**Sahel Mohammad Iqbal**



Aalto University publication series  
Doctoral Theses 106/2026

# **Statistical methods for sequential decision making and inference**

Sahel Mohammad Iqbal

A doctoral thesis completed for the degree of Doctor of Science (Technology) to be defended, with the permission of the Aalto University School of Electrical Engineering, at a public examination held at the lecture hall AS2 (Maarintie 8, Espoo) on 08 May 2026 at 12pm EEST. It will also be broadcast online via the link <https://aalto.zoom.us/j/64049287751>.

Aalto University  
School of Electrical Engineering  
Department of Electrical Engineering and Automation  
Sensor Informatics and Medical Technology

**Supervising professor**

Professor Simo Särkkä, Aalto University, Finland

**Thesis advisors**

Professor Simo Särkkä, Aalto University, Finland

**Preliminary examiners**

Professor Jimmy Olsson, KTH Royal Institute of Technology, Sweden

Professor Youssef M. Marzouk, Massachusetts Institute of Technology, USA

**Opponent**

Professor Jimmy Olsson, KTH Royal Institute of Technology, Sweden

Aalto University publication series

Doctoral Theses 106/2026

© Sahel Mohammad Iqbal

ISBN 978-952-64-3135-2 (paperback)

ISBN 978-952-64-3134-5 (PDF)

ISSN 1799-4934 (print)

ISSN 1799-4942 (online)

<https://urn.fi/URN:ISBN:978-952-64-3134-5>

PunaMusta Oy

Helsinki 2026

---

**Author** Sahel Mohammad Iqbal

---

**Name of the doctoral thesis** Statistical methods for sequential decision making and inference

---

**Article-based thesis**

---

**Number of pages** 148

---

**Keywords** Bayesian experimental design, Stochastic optimal control, Sequential Monte Carlo

---

Sequential decision making under uncertainty is a fundamental challenge across engineering and science. In these problems, decision makers must infer hidden states of the world and take actions based on them to maximize some expected utility over a horizon. Finding optimal behavior is notoriously difficult: each decision requires reasoning about (i) the immediate utility obtained, (ii) how the decision's outcome improves knowledge of the hidden state, and (iii) how this improved state estimate could yield better utility over remaining decisions. Most existing algorithms for sequential decision making rely on restrictive approximations to make this otherwise computationally intractable problem tractable.

This thesis develops a unified inference and learning framework that addresses these challenges by leveraging the connection between stochastic optimal control and probabilistic inference. Using this framework, we build algorithms that reason over future decisions and observations without resorting to common approximations, and amortize this behavior into history-dependent policies that can be deployed in real-time systems. We show that different sequential decision problems---namely optimal control of Markov decision processes (MDPs) and partially observed MDPs, and sequential Bayesian experimental design (BED)---can be framed as maximum likelihood estimation in Feynman-Kac models, providing a unified conceptual framework for understanding superficially distinct problems. We then solve these decision problems by introducing nested sequential Monte Carlo (SMC) algorithms that efficiently plan in the space of decisions and observations. The generality of this SMC framework enables applications to nonlinear, non-Gaussian, and non-Markovian settings.

Beyond decision making, we extend this probabilistic inference perspective to numerical computation. We develop a novel Bayesian probabilistic numerical algorithm for solving time-dependent nonlinear partial differential equations (PDEs). Collectively, this thesis provides a unified theoretical perspective and a suite of advanced computational tools for optimal decision making and inference in dynamical systems.



# Preface

The research presented in this thesis was conducted under the supervision of Prof. Simo Särkkä between November 2022 and October 2025 at the Department of Electrical Engineering and Automation of Aalto University, Finland. I gratefully acknowledge financial support from Business Finland and the Academy of Finland.

There are three people who have played an outsized role both in shaping this thesis and in my development as a researcher, whom I wish to thank first. I would like to express my deepest gratitude to Simo Särkkä for giving me the opportunity to pursue a PhD, and for the freedom to work on the problems that truly interested me. Your work ethic and drive continue to inspire me to become a better researcher, and I thank you for fostering a fantastic research environment. I am forever grateful to Hany Abdulsamad for being the greatest mentor and friend that I could have wished for during my PhD. This thesis is as much your success as it is mine. I am also deeply grateful to Adrien Corenflos for teaching me so many things I didn't know and pushing me to do so many things I didn't think I could.

I would like to thank my other great friends at Aalto: Christos Merktas for being my first friend at the office and for making me feel welcome, and Casian Iacob for your sense of humor and your company. It's impossible to pick between F308 and F402 as the more fun office. I am also grateful to all my other wonderful colleagues who made Aalto a great place to be: Cristian, Fatemeh, Hassan, Iqbal, Ivan, Kundan, Lauri, Lisa, Sara, Teemu, and Zaeed. Beyond the university, I am grateful to Akshara, Alok, Inshu, Sid, and Shraman for their friendship through the bleak Finnish winters.

The greatest thanks go to my friends and family back home for their love and companionship. To my dear friends Anagha, Anamika, Pinky, Rifa, Saran, and Sharun—staying sane throughout my PhD would have been impossible if not for you. To my brother Chachu and my cousins Fadi, Fawa, Firu, Reenu, and Sabeel—with whom I have had and continue to have a lot of fun. To Tanishka, for being my greatest companion and cheerleader throughout the three years of my PhD. And finally, to my parents, for their love and for giving me the freedom to choose my own path in life.

Preface

Helsinki and Thrissur, April 9, 2026,

Sahel Mohammad Iqbal

# Contents

<b>Preface</b>	<b>1</b>
<b>Contents</b>	<b>3</b>
<b>List of Publications</b>	<b>5</b>
<b>Author's Contribution</b>	<b>7</b>
<b>List of Figures</b>	<b>9</b>
<b>Abbreviations</b>	<b>11</b>
<b>Symbols</b>	<b>13</b>
<b>1. Introduction</b>	<b>15</b>
<b>1.1 Thesis outline</b> . . . . .	16
<b>2. Statistical inference in sequential models</b>	<b>19</b>
<b>2.1 Markov models</b> . . . . .	20
2.1.1 State-space models . . . . .	20
2.1.2 Filtering and smoothing . . . . .	22
<b>2.2 Exact inference in linear-Gaussian SSMs</b> . . . . .	24
2.2.1 Kalman filter . . . . .	24
2.2.2 Pathwise and marginal smoothers . . . . .	24
<b>2.3 Gaussian-approximated filters and smoothers</b> . . . . .	25
2.3.1 The iterated extended Kalman smoother . . . . .	26
<b>2.4 Feynman-Kac models</b> . . . . .	27
<b>3. Sequential Decision Making as Inference</b>	<b>29</b>
<b>3.1 Stochastic optimal control of Markov decision processes</b>	<b>29</b>
3.1.1 The dynamic programming solution . . . . .	30
3.1.2 The inference problem . . . . .	31
3.1.3 Policy optimization . . . . .	33

<b>3.2</b>	<b>An inference and learning framework for sequential decision making</b> . . . . .	34
<b>3.3</b>	<b>Partially observable Markov decision processes</b> . . . . .	36
3.3.1	Background . . . . .	36
3.3.2	The inference formulation . . . . .	38
<b>3.4</b>	<b>Bayesian experimental design</b> . . . . .	39
3.4.1	Background . . . . .	39
3.4.2	Designing sequential experiments . . . . .	41
3.4.3	The inference formulation . . . . .	42
<b>4.</b>	<b>Particle Approximations in General Sequential Models</b>	<b>45</b>
<b>4.1</b>	<b>The static setting</b> . . . . .	47
4.1.1	The Monte Carlo method . . . . .	47
4.1.2	Importance sampling . . . . .	47
4.1.3	The effective sample size . . . . .	48
<b>4.2</b>	<b>The sequential setting</b> . . . . .	48
4.2.1	Sequential importance sampling . . . . .	49
4.2.2	Resampling . . . . .	49
4.2.3	Sequential Monte Carlo . . . . .	50
4.2.4	Sequential Monte Carlo Smoother . . . . .	51
<b>4.3</b>	<b>Advanced SMC methods</b> . . . . .	52
4.3.1	The resample-move algorithm . . . . .	52
4.3.2	Iterated Batch Importance Sampling . . . . .	53
<b>5.</b>	<b>Bayesian ODE Solvers and Parallel-in-Time State Estimation</b>	<b>55</b>
<b>5.1</b>	<b>ODE solving as Bayesian inference</b> . . . . .	55
5.1.1	Stochastic differential equation priors . . . . .	56
5.1.2	The state-space model formulation . . . . .	57
<b>5.2</b>	<b>Temporal parallelization of Bayesian smoothers</b> . . . . .	58
5.2.1	Prefix-sums . . . . .	58
5.2.2	Bayesian filtering and smoothing as prefix-sums . . . . .	60
<b>6.</b>	<b>Summary and Discussion</b>	<b>63</b>
<b>6.1</b>	<b>Publication I: Nesting particle filters for experimental design in dynamical systems</b> . . . . .	63
<b>6.2</b>	<b>Publication II: Sequential Monte Carlo for policy optimization in continuous POMDPs</b> . . . . .	64
<b>6.3</b>	<b>Publication III: Parallel-in-time probabilistic solutions for time-dependent nonlinear partial differential equations</b> . . . . .	65
<b>6.4</b>	<b>Publication IV: Recursive nested filtering for efficient amortized Bayesian experimental design</b> . . . . .	66
	<b>References</b>	<b>69</b>
	<b>Publications</b>	<b>77</b>

# List of Publications

This thesis consists of an overview and of the following publications which are referred to in the text by their Roman numerals. The superscript "\*" denotes equal contribution.

- I** Sahel Iqbal, Adrien Corenflos, Simo Särkkä, Hany Abulsamad. Nesting particle filters for experimental design in dynamical systems. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, Vienna, Austria, Pages 21047-21068, July 2024.
- II** Hany Abulsamad\*, Sahel Iqbal\*, Simo Särkkä. Sequential Monte Carlo for policy optimization in continuous POMDPs. In *Proceedings of the 39th Annual Conference on Neural Information Processing Systems (NeurIPS)*, San Diego, United States of America, Pages 1-22, December 2025.
- III** Sahel Iqbal, Hany Abdulsamad, Tripp Cator, Ulisses Braga-Neto, Simo Särkkä. Parallel-in-time probabilistic solutions for time-dependent nonlinear partial differential equations. In *Proceedings of the IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, London, United Kingdom, Pages 1-6, September 2024.
- IV** Sahel Iqbal, Hany Abdulsamad, Sara Pérez-Vieites, Simo Särkkä, Adrien Corenflos. Recursive nested filtering for efficient amortized Bayesian experimental design. In *Neural Information Processing Systems Workshop on Bayesian Decision-Making and Uncertainty*, Vancouver, Canada, Pages 1-14, December 2024.



# Author's Contribution

## **Publication I: “Nesting particle filters for experimental design in dynamical systems”**

The original idea for the article was conceived by Hany Abdulsamad and developed jointly with Sahel Iqbal and Adrien Corenflos. Sahel Iqbal developed and implemented Inside-Out SMC<sup>2</sup> with Adrien Corenflos's guidance, and the proof of its consistency is due to Adrien Corenflos. The experiments are due equally to Sahel Iqbal and Hany Abdulsamad. Hany Abdulsamad supervised the project. Sahel Iqbal wrote the initial manuscript, and all authors contributed to revisions. Simo Särkkä reviewed and validated the technical details.

## **Publication II: “Sequential Monte Carlo for policy optimization in continuous POMDPs”**

The original idea and motivation are due to Hany Abdulsamad. The methodology, proofs, and experiments were developed jointly by Hany Abdulsamad and Sahel Iqbal, with equal contribution. Sahel Iqbal and Hany Abdulsamad drafted the manuscript. Simo Särkkä provided regular feedback through weekly meetings, contributed valuable insights on the setting and background literature, and proofread the manuscript.

## **Publication III: “Parallel-in-time probabilistic solutions for time-dependent nonlinear partial differential equations”**

The writing is due to Sahel Iqbal, and the experimental evaluation is due equally to Sahel Iqbal and Hany Abdulsamad, with help from Tripp Cator. The original idea for the article is due to Simo Särkkä.

**Publication IV: “Recursive nested filtering for efficient amortized Bayesian experimental design”**

Adrien Corenflos and Hany Abdulsamad initially conceived the idea for this article, with Adrien Corenflos and Sahel Iqbal then developing the methodology. Sahel Iqbal proved the consistency of the method following Adrien Corenflos’s suggestions. The code implementation and experiments were carried out jointly by Sahel Iqbal and Hany Abdulsamad. Sahel Iqbal took the lead in writing the article, with valuable help from Adrien Corenflos and Hany Abdulsamad. Sara Pérez-Vieites and Simo Särkkä provided helpful discussions and feedback on the manuscript.

# List of Figures

2.1	Graphical model of an SSM. Latent states $X_t$ generate observations $Y_t$ . . . . .	21
-----	--	----



# Abbreviations

**BED** Bayesian experimental design

**BPN** Bayesian probabilistic numerical

**DM** Decision maker

**DP** Dynamic programming

**e.g.** *Exempli gratia* (for example)

**EIG** Expected information gain

**EKF** Extended Kalman filter

**ESS** Effective sample size

**FK** Feynman-Kac

**GP** Gaussian process

**GPU** Graphics processing unit

**i.e.** *Id est* (that is)

**IEKS** Iterated extended Kalman smoother

**IS** Importance sampling

**LGSSM** Linear-Gaussian state-space model

**MAP** Maximum a posteriori

**MDP** Markov decision process

**MLE** Maximum likelihood estimate

**NASA** National Aeronautics and Space Administration

**NPF** Nested particle filter

Abbreviations

**ODE** Ordinary differential equation

**PDE** Partial differential equation

**POMDP** Partially observable Markov decision process

**RTS** Rauch-Tung-Striebel

**SDE** Stochastic differential equation

**SIS** Sequential importance sampling

**SMC** Sequential Monte Carlo

**SOC** Stochastic optimal control

**SSM** State-space model

# Symbols

## *Spaces and sequences:*

$\mathbb{N}$	The set of natural numbers $\{1, 2, \dots\}$ .
$\mathbb{N}_0$	The set of non-negative integers $\{0\} \cup \mathbb{N}$ .
$\mathbb{R}^d$	The $d$ -dimensional Euclidean space.
$\mathbb{R}_+$	The set of non-negative real numbers.
$x_{m:n}$	A sequence of values $(x_m, x_{m+1}, \dots, x_n)$ for $m \leq n$ .
$x^{m:n}$	A sequence of values $(x^m, x^{m+1}, \dots, x^n)$ for $m \leq n$ .
$X$	The space in which latent variables, or hidden states of the world, take values. Unless otherwise specified, $X$ is assumed to be a subset of a Euclidean space equipped with the Borel $\sigma$ -algebra.
$Y$	The space in which observations take values. Unless otherwise specified, $Y$ is assumed to be a subset of a Euclidean space equipped with the Borel $\sigma$ -algebra.
$[m : n]$	The sequence $\{m, m + 1, \dots, n\}$ , for $m, n \in \mathbb{N}_0$ , $m \leq n$ .

## *Measures and random variables:*

$\mathbb{E}_\mu[X]$	Expectation of a random variable $X$ with law $\mu$ . We omit the subscript and write $\mathbb{E}[X]$ if the law of $X$ is clear from the context.
$\mathbb{E}[X   Y]$	Conditional expectation of the random variable $X$ given the random variable $Y$ .
$\mathbb{H}[X]$	Entropy of a random variable $X$ , $\mathbb{H}[X] := \mathbb{E}[-\log \mu(X)]$ .
$\mathcal{N}(b, C)$	Gaussian distribution with mean $b$ and covariance $C$ .

Symbols

$\mathcal{N}(a; b, C)$	Density of a Gaussian distribution with mean $b$ and covariance $C$ evaluated at $a$ .
$\mathcal{P}(X)$	The set of probability measures on a space $X$ , with the measurable space typically clear from the context.
$X \sim \mu$	A random variable $X$ with distribution $\mu$ .
$\text{Var}_\mu[X]$	The variance of $X$ , $\mathbb{E}_\mu[(X - \mathbb{E}_\mu[X])^2]$ .
$\delta_a(dx)$	The Dirac measure at a point $a$ .
$\mu(dx)$	A measure on a measurable space $(X, \mathcal{X})$ . The notation is equivalent to $d\mu(x)$ , corresponding to the measure of an infinitesimal region around $x$ .
$\mu(f)$	Alternative notation for $\mathbb{E}[f(X)]$ for some integrable function $f$ , where $X \sim \mu$ .

*Miscellaneous:*

$\text{argmax}$	Arguments of the maxima. For any function $f : X \rightarrow \mathbb{R}$ , it is defined as $\text{argmax}_{x \in X} f(x) := \{x \in X : f(x') \leq f(x) \text{ for all } x' \in X\}$ .
$\text{argmin}$	Arguments of the minima. For any function $f : X \rightarrow \mathbb{R}$ , it is defined as $\text{argmin}_{x \in X} f(x) := \{x \in X : f(x) \leq f(x') \text{ for all } x' \in X\}$ .
$A^{-1}$	Inverse of a matrix $A$ .
$A^\top$	Transpose of a matrix $A$ .
$\mathcal{O}(f(n))$	The set of functions $\{g(n) : \exists c, n_0 > 0 \text{ such that } \forall n \geq n_0, 0 \leq g(n) \leq cf(n)\}$ . Used to denote an asymptotic upper bound for the computational and time complexity of algorithms.
$\nabla f(x)$	Jacobian matrix of a vector-valued function $f : X \rightarrow \mathbb{R}^d$ evaluated at a point $x \in X$ .

# 1. Introduction

What connects the problem of landing a spacecraft on the Moon with measuring people's willingness to wait for bigger rewards? For the lunar module from NASA's Apollo program, engineers needed to design a guidance system that continuously estimated the module's position, velocity, and orientation from noisy sensor measurements, and then used this information to compute the thrust magnitude and nozzle direction needed to safely land on the lunar surface (Suddath et al., 1967). By contrast, when psychologists study temporal discounting, they pose questions like "Would you prefer \$10 today or \$100 in a month?", update their beliefs about a participant's preferences after each response, and iteratively refine their questions until they can reliably estimate the underlying discount rate (Vincent, 2016).

Both scenarios exemplify problems of *sequential inference and decision making under uncertainty*. In each case, a *decision maker* (DM) maintains beliefs about some hidden state of the world—the spacecraft's true trajectory, or a person's discount rate—and must choose actions with the intention of advancing toward a goal. After each action, new observations arrive: sensor readings, or a participant's choice. The decision maker then updates their beliefs to be consistent with the observations through an inference procedure, and the process repeats.

The examples above highlight the intimate connection between inference and decision making. A better estimate of the true state can help the DM choose superior actions, while certain actions could reveal observations that lead to a more accurate state estimate. However, the connection between (statistical) inference and decision making runs even deeper. A DM's beliefs can be understood as subjective probabilities that obey coherent updating rules *if* the DM makes consistent decisions under uncertainty (Savage, 1954). Moreover, statistical inference itself can be formulated as a decision problem (Bernardo and Smith, 1994), while conversely, decision problems can be cast as statistical inference (Kálmán, 1960; Kueck et al., 2009).

This latter connection between decision making and inference was first observed by Kálmán, who discovered an exact duality between certain optimal control and state estimation problems (Kálmán, 1960). More recent work

has generalized this idea by introducing auxiliary variables that represent goal satisfaction and conditioning on them, transforming the search for optimal decisions into posterior inference (Toussaint and Storkey, 2006; Rawlik, 2013). In informal terms, we ask: "*Assuming the goal has been achieved, what decisions must have been made?*".

Building on this connection, this thesis develops Bayesian inference methods (Bernardo and Smith, 1994), with an emphasis on methods based on random sampling (Doucet et al., 2001), for solving decision problems and related computational problems. The central theme is to formulate sequential decision making under uncertainty as a problem of statistical estimation, where optimal policies or control strategies emerge from maximizing likelihoods or posterior probabilities. This inferential viewpoint is then extended beyond decision problems to numerical analysis, by showing that solving a partial differential equation can likewise be interpreted as an inference problem. Across these settings, the thesis develops algorithms that unify learning, estimation, and control under a single probabilistic perspective.

## 1.1 Thesis outline

The thesis is organized as follows. Chapters 2 through 5 provide the necessary background for understanding the contributions in Publications I to IV. Chapter 6 discusses each publication in detail, and the publications themselves are appended at the end. The contents of each chapter are as follows.

Chapter 2 reviews the foundations of statistical inference for discrete-time sequential models. We introduce the two model classes central to this thesis—state-space models (SSMs) and Feynman-Kac (FK) models—and present the general Bayesian filtering and smoothing recursions for state estimation, together with algorithms for computing Gaussian state estimates in both linear and nonlinear SSMs.

Chapter 3 develops a unified inference-based formulation of sequential decision making. We begin with stochastic optimal control for Markov decision processes (MDPs), then present a general procedure applicable across a range of decision problems. We also introduce partially observable MDPs (POMDPs) and sequential Bayesian experimental design.

Chapter 4 turns to flexible, non-Gaussian inference via particle approximations. We present Monte Carlo methods for approximating FK models, laying the groundwork for the methodological contributions of the thesis.

Chapter 5 applies the inferential perspective to numerical analysis. It outlines Bayesian ODE solvers based on filtering and smoothing, and introduces temporal-parallelization schemes that enable efficient GPU-accelerated inference.

Chapter 6 summarizes the contributions of the four publications included

in the thesis, relates them to the topics developed in the preceding chapters, and discusses directions for future research.



## 2. Statistical inference in sequential models

The goal of statistical inference is to characterize uncertainty about unknown quantities using observed data. As discussed in the previous chapter, inference often serves both as a prerequisite for decision making and as a tool for solving decision problems. Here we provide a concise introduction to statistical inference from a *Bayesian* perspective (Bernardo and Smith, 1994), which we adopt throughout the remainder of the thesis.

Let  $X$  denote the possible hidden states of the world and  $Y$  the set of observations or data. A decision maker starts with a *prior* density  $p(x)$ , which encodes their initial beliefs about the hidden state as probabilities over different realizations  $x \in X$ . The DM also has a *likelihood function*  $p(y | x)$ , a conditional probability density function that describes the probability of observing  $y \in Y$  if the hidden state is  $x$ . Then, if the DM observes data  $y$ , Bayes' rule provides the rational way to update their beliefs, yielding a *posterior* (Savage, 1954)

$$p(x | y) = \frac{p(y | x)p(x)}{p(y)}.$$

The normalizing constant in the denominator,  $p(y) = \int p(y | x)p(x)dx$ , is known as the *marginal likelihood*.

Many quantities of interest in Bayesian statistics take the form of posterior expectations  $\int f(x)p(x | y)dx$  for suitable functions  $f$ , which among other things are used to summarize predictions and inform decisions (Berger, 1985). In practice, for many complex models of interest, the marginal likelihood and hence the posterior are not available in closed form. Computing estimates of the posterior and of posterior expectations is therefore the central challenge of Bayesian statistics (Martin et al., 2023).

While the posterior  $p(x | y)$  is the main object of study in Bayesian inference, two point estimates of the hidden state are often used and relevant for this thesis. The first of these is a mode of the posterior,  $x^* \in \operatorname{argmax}_x p(x | y)$ , known as the *maximum a posteriori* (MAP) estimate. The second, called a *maximum likelihood estimate* (MLE, Stigler, 2007), is defined as  $x^* \in \operatorname{argmax}_x p(y | x)$ . Both the MAP estimate and the MLE are frequently used

in the setting of parameter inference, where the state of the world  $x$  is seen as a parameter of the likelihood function  $p(y | x)$ .

In the remainder of this chapter, we focus on *sequential* (time-indexed) models, where uncertainty evolves and data arrive over time. We start by reviewing the basic concepts of Markov chains and state-space models (SSMs) in Section 2.1. We then cover exact inference in linear-Gaussian SSMs in Section 2.2 and discuss Gaussian approximations for state estimates in nonlinear SSMs in Section 2.3. Finally, in Section 2.4, we introduce a very general class of sequential models known as Feynman-Kac models.

**Notation and conventions.** We denote random variables by uppercase letters and their realizations by corresponding lowercase letters, e.g., random variable  $X$  has a realization  $x$ . We write  $X \sim \mu$  to mean random variable  $X$  has distribution (or density)  $\mu$ , and we denote the expectation of a function  $f$  by  $\mathbb{E}_\mu[f(X)]$ , with the subscript  $\mu$  omitted when the distribution of  $X$  is clear from the context. We only consider continuous state and observation spaces, so throughout the thesis we assume  $X \subseteq \mathbb{R}^{d_x}$  and  $Y \subseteq \mathbb{R}^{d_y}$  for  $d_x, d_y \in \mathbb{N}$ . For  $m, n \in \mathbb{N}$ ,  $m \leq n$ , we denote  $[m : n] := \{m, m + 1, \dots, n\}$  and  $x_{m:n} := (x_m, x_{m+1}, \dots, x_n)$ .

## 2.1 Markov models

We begin by introducing Markov chains—stochastic processes in which the current state summarizes all past information for predicting the next.

**Definition 2.1** (Markov chain). *A sequence of random variables  $(X_t)_{t=0}^T := (X_0, X_1, \dots, X_T)$  is called a discrete-time Markov chain if its distribution  $p$  factorizes as*

$$p(x_{0:T}) = p_0(x_0) \prod_{t=1}^T p_t(x_t | x_{t-1}), \quad (2.1)$$

where  $p_0$  is an initial distribution and  $p_t(x_t | x_{t-1})$  are conditional probability density functions for  $t \in \{1, \dots, T\}$ .

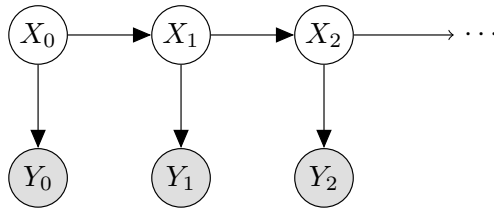
The factorization (2.1) can be equivalently stated as the condition

$$p(x_t | x_{0:t-1}) = p_t(x_t | x_{t-1}), \quad t \geq 1, \quad (2.2)$$

which is known as the Markov property. As evident from (2.2), Markov chains are memoryless processes, meaning the next state is conditionally independent of the past given the current state.

### 2.1.1 State-space models

A related concept to Markov chains is that of state-space models (SSMs), also known as hidden Markov models (HMMs, Cappé et al., 2005; Särkkä



**Figure 2.1.** Graphical model of an SSM. Latent states  $X_t$  generate observations  $Y_t$ .

and Svensson, 2023). SSMs are a class of probabilistic models that is ubiquitous in many fields including statistics, biology, and finance. They are used to model dynamical phenomena where the underlying state is Markovian, but the state is hidden and we only have access to noisy observations at each time step; see Figure 2.1.

**Definition 2.2** (State-space model). *A state-space model is a bivariate stochastic process  $(X_t, Y_t)_{t=0}^T$  taking values in  $(X \times Y)^{T+1}$  with joint density*

$$p(x_{0:T}, y_{0:T}) = p_0(x_0) \prod_{t=1}^T p_t(x_t | x_{t-1}) \prod_{s=0}^T g_s(y_s | x_s).$$

*The process  $(X_t)_{t=0}^T$  is a Markov chain with initial density  $p_0$  and transition densities  $\{p_t\}_{t \geq 1}$ , and the observations  $Y_{0:T}$  are conditionally independent given  $X_{0:T}$ , with  $Y_t$  depending only on  $X_t$  through the conditional density function  $g_t(y_t | x_t)$ .*

For brevity, and to unify the presentation with a wider class of models introduced in Section 2.4, we introduce the notation

$$q_t(x_{0:s}) := p(x_{0:s} | y_{0:t}) \propto p_0(x_0) \prod_{k=1}^s p_k(x_k | x_{k-1}) \prod_{l=0}^t g_l(y_l | x_l), \quad (2.3)$$

for  $0 \leq t \leq s \leq T$ . For  $s < t$ , we can define  $q_t(x_{0:s})$  by integrating out the future states from the joint distribution:

$$q_t(x_{0:s}) := \int q_t(x_{0:t}) dx_{s+1:t}, \quad 0 \leq s < t \leq T.$$

We also introduce related notation for the marginals,  $q_t(x_s) := \int q_t(x_{0:s}) dx_{0:s-1}$ . Then the three main inference problems in SSMs concern the following distributions:

- *The filtering distribution,  $q_t(x_t) = p(x_t | y_{0:t})$ , is the posterior distribution of the current state  $x_t$  given observations up to time  $t$ .*
- *The marginal smoothing distribution,  $q_T(x_t) = p(x_t | y_{0:T})$ , is the posterior distribution of  $x_t$  conditioned on all observations, past and future.*

- *The pathwise smoothing distribution,  $q_T(x_{0:T}) = p(x_{0:T} | y_{0:T})$ , is the joint posterior over complete trajectories  $x_{0:T}$  given all observations.*

A fourth problem of interest is parameter estimation: when the dynamics or measurement models depend on unknown parameters  $\theta$ , we may seek an MLE  $\theta^* \in \arg \max_{\theta} p(y_{0:T} | \theta)$ , where  $p(y_{0:T} | \theta)$  is the marginal likelihood.

### 2.1.2 Filtering and smoothing

The temporal structure of the distributions of interest in SSMs (2.3) enables efficient recursive computation of the filtering and smoothing distributions. The equations for computing the filtering distributions recursively are given by the following theorem.

**Theorem 2.3** (Bayesian filtering equations). *If the filtering distribution  $q_{t-1}(x_{t-1})$  at time  $t-1$  is known, then the filtering distribution at time  $t$  can be computed in two steps:*

1. Prediction step: *Predict the distribution of  $x_t$  using the dynamic model,*

$$q_{t-1}(x_t) = \int p_t(x_t | x_{t-1}) q_{t-1}(x_{t-1}) dx_{t-1}. \quad (2.4)$$

2. Correction step: *Modify the predictive distribution to take the current observation into account,*

$$q_t(x_t) \propto q_{t-1}(x_t) g_t(y_t | x_t). \quad (2.5)$$

*Proof.* For (2.4), note that

$$q_{t-1}(x_t) = p(x_t | y_{0:t-1}) = \int p(x_t | x_{t-1}, y_{0:t-1}) p(x_{t-1} | y_{0:t-1}) dx_{t-1},$$

and we use the fact that  $x_t$  is conditionally independent of  $y_{0:t-1}$  given  $x_{t-1}$ . Equation (2.5) is Bayes' rule.  $\square$

This enables computing the filtering distributions in an online fashion, i.e., as observations arrive and without needing to reprocess all past observations at each time step.

Once we have the filtering distributions, the pathwise and marginal smoothing distributions can also be computed recursively. Using the chain rule of probability, we may write

$$q_T(x_{0:T}) = q_T(x_T) \prod_{t=0}^{T-1} q_T(x_t | x_{t+1:T}), \quad (2.6)$$

where  $q_T(x_t | x_{t+1:T}) := p(x_t | x_{t+1:T}, y_{0:T})$ . The terminal filtering distribution  $q_T(x_T)$  is also the marginal smoothing distribution at time  $T$ . In addition, the expression (2.6) can be further simplified using the following theorem.

**Theorem 2.4** (Markov property of the backward process). *Let  $(X_t)_{t=0}^T$  be the state process of the SSM defined in Definition 2.2. The time-reversed process conditioned on the observations  $y_{0:T}$ , denoted by the conditional density  $q_T(x_t | x_{t+1:T})$  appearing in (2.6), forms a Markov chain. Specifically:*

$$q_T(x_t | x_{t+1:T}) = q_t(x_t | x_{t+1}), \quad t \in \{0, \dots, T-1\}.$$

*Proof.* Using Bayes' rule and the conditional independence properties of the state-space model, we write:

$$\begin{aligned} q_T(x_t | x_{t+1:T}) &= p(x_t | x_{t+1:T}, y_{0:T}) \\ &= \frac{p(x_{t+1:T}, y_{t+1:T} | x_t, y_{0:t}) p(x_t | y_{0:t})}{p(x_{t+1:T}, y_{t+1:T} | y_{0:t})} \\ &= \frac{p(x_{t+1:T}, y_{t+1:T} | x_t) p(x_t | y_{0:t})}{p(x_{t+1:T}, y_{t+1:T} | y_{0:t})} \\ &= \frac{p(x_{t+2:T}, y_{t+1:T} | x_{t+1}) p_{t+1}(x_{t+1} | x_t) p(x_t | y_{0:t})}{p(x_{t+2:T}, y_{t+1:T} | x_{t+1}) p(x_{t+1} | y_{0:t})} \\ &= \frac{p_{t+1}(x_{t+1} | x_t) p(x_t | y_{0:t})}{p(x_{t+1} | y_{0:t})} \\ &= p(x_t | x_{t+1}, y_{0:t}) =: q_t(x_t | x_{t+1}). \end{aligned}$$

□

The following theorem describes how to compute  $q_t(x_t | x_{t+1})$  recursively backwards using quantities computed in the filtering pass (Theorem 2.3).

**Theorem 2.5** (Bayesian smoothing equations). *The smoothed backward transition kernel  $q_t(x_t | x_{t+1})$  is given by*

$$q_t(x_t | x_{t+1}) = q_t(x_t) \frac{p_{t+1}(x_{t+1} | x_t)}{q_t(x_{t+1})}. \quad (2.7)$$

*Then, given the marginal smoothing distribution  $q_T(x_{t+1})$  at time  $t+1$ , the distribution  $q_T(x_t)$  can be computed as*

$$q_T(x_t) = \int q_t(x_t | x_{t+1}) q_T(x_{t+1}) dx_{t+1}. \quad (2.8)$$

*Proof.* See Särkkä and Svensson (2023, Thm. 12.1). □

For general SSMs, neither the Bayesian filtering nor smoothing equations may be available in closed form, and we have to resort to approximations for each of those steps. Nevertheless, before considering such approximations, we briefly discuss the special case when the dynamic and measurement models are affine and Gaussian, which admits closed-form computation of the filtering and smoothing equations.

## 2.2 Exact inference in linear-Gaussian SSMs

Let  $\mathcal{N}(a; b, C)$  denote the density of a Gaussian distribution with mean  $b$  and covariance  $C$  evaluated at a point  $a$ . A *linear-Gaussian SSM* (LGSSM) is a state-space model in which the dynamic and measurement models are affine and Gaussian, with Gaussian initial condition:

$$\begin{aligned} p_0(x_0) &= \mathcal{N}(x_0; m_0, P_0), \\ p_t(x_t | x_{t-1}) &= \mathcal{N}(x_t; A_{t-1}x_{t-1} + b_{t-1}, Q_{t-1}), \\ g_t(y_t | x_t) &= \mathcal{N}(y_t; C_t x_t + d_t, R_t). \end{aligned}$$

Here,  $(m_0, P_0)$  are the initial mean and covariance,  $(A_{t-1}, b_{t-1}, Q_{t-1})$  are the *transition matrix*, *transition offset*, and *transition covariance* respectively, and  $(C_t, d_t, R_t)$  are the *observation matrix*, *observation offset* and *observation covariance* respectively. In this case, the equations in Theorems 2.3 and 2.5 can be solved exactly, and also yield Gaussian distributions. We detail this computation next.

### 2.2.1 Kalman filter

The solution to the Bayesian filtering equations for LGSSMs is known as the *Kalman filter* (Kálmán, 1960). Assume we have already computed  $q_{t-1}(x_{t-1}) = \mathcal{N}(x_{t-1}; m_{t-1}^-, P_{t-1}^-)$ . The Kalman filter provides closed-form solutions for the prediction step (2.4):

$$q_{t-1}(x_t) = \mathcal{N}(x_t; m_t^-, P_t^-), \quad (2.9)$$

$$m_t^- = A_{t-1}m_{t-1}^- + b_{t-1}, \quad P_t^- = A_{t-1}P_{t-1}^-A_{t-1}^\top + Q_{t-1},$$

and the update step (2.5):

$$q_t(x_t) = \mathcal{N}(x_t; m_t, P_t), \quad (2.10)$$

$$m_t = m_t^- + K_t v_t, \quad P_t = P_t^- - K_t S_t K_t^\top,$$

$$v_t = y_t - C_t m_t^- - d_t, \quad S_t = C_t P_t^- C_t^\top + R_t, \quad K_t = P_t^- C_t^\top S_t^{-1}.$$

The Kalman filter also provides the marginal likelihood increments  $p(y_t | y_{0:t-1})$ ,

$$p(y_t | y_{0:t-1}) = \mathcal{N}(y_t; C_t m_t^- + d_t, S_t).$$

These formulas follow from standard properties of Gaussian distributions and a derivation can be found in Särkkä and Svensson (2023, Theorem 6.6).

### 2.2.2 Pathwise and marginal smoothers

We now describe how to compute the pathwise and marginal smoothing distribution for LGSSMs. The backward transition kernel  $q_t(x_t | x_{t+1})$  in (2.7)

has the form

$$q_t(x_t | x_{t+1}) = \mathcal{N}(x_t; \mathbf{G}_t x_{t+1} + f_t, \mathbf{H}_t),$$

$$\mathbf{G}_t = \mathbf{P}_t \mathbf{A}_t^\top [\mathbf{P}_{t+1}^-]^{-1}, \quad f_t = m_t - \mathbf{G}_t m_{t+1}^-, \quad \mathbf{H}_t = \mathbf{P}_t - \mathbf{G}_t \mathbf{P}_{t+1}^- \mathbf{G}_t^\top,$$

where  $(m_t, \mathbf{P}_t)$  is the filtering mean and covariance at time  $t$  (2.10) and  $(m_{t+1}^-, \mathbf{P}_{t+1}^-)$  are the predictive mean and covariance at time  $t+1$  (2.9). Furthermore, if we know that  $q_T(x_{t+1}) = \mathcal{N}(x_{t+1}; m_{t+1}^s, \mathbf{P}_{t+1}^s)$ , the marginalization in (2.8) can be performed in closed form to obtain the distribution  $q_T(x_t)$ :

$$q_T(x_t) = \mathcal{N}(x_t; m_t^s, \mathbf{P}_t^s),$$

$$m_t^s = m_t + \mathbf{G}_t [m_{t+1}^s - m_{t+1}^-], \quad \mathbf{P}_t^s = \mathbf{P}_t + \mathbf{G}_t [\mathbf{P}_{t+1}^s - \mathbf{P}_{t+1}^-] \mathbf{G}_t^\top.$$

This procedure to obtain the marginal smoothing distributions in LGSSMs is known as the *Rauch-Tung-Striebel (RTS) smoother* (Rauch et al., 1965), or sometimes simply the *Kalman smoother*.

### 2.3 Gaussian-approximated filters and smoothers

In many practical cases of interest, one or both of the dynamic and observation models may be nonlinear. For such nonlinear SSMS, the Bayesian filtering and smoothing equations may not admit closed-form solutions, and we have to resort to approximations. We develop one such class of approximations in this section, that of approximating distributions of interest as Gaussians. We restrict ourselves to the case where the dynamic and measurement models have Gaussian noise, though the methods themselves are applicable for even more general settings (see, e.g., Särkkä and Svensson, 2023, Ch. 10).

Suppose the dynamic and observation models are

$$p_t(x_t | x_{t-1}) = \mathcal{N}(x_t; f(x_{t-1}), \mathbf{Q}_{t-1}),$$

$$g_t(y_t | x_t) = \mathcal{N}(y_t; h(x_t), \mathbf{R}_t),$$
(2.11)

where  $f: X \rightarrow X$  and  $h: X \rightarrow Y$  are general functions, and  $\mathbf{Q}_{t-1}$  and  $\mathbf{R}_t$  are the transition and observation covariances respectively.  $f$  and  $h$  may also depend on the time step  $t$ , but we omit this from the notation for simplicity. The algorithms developed in Section 2.2 suggest a straightforward way to approximate the filtering and smoothing distributions with Gaussians:

1. Form affine approximations to  $f$  and  $h$  to obtain a linear-Gaussian approximation to (2.11).
2. Use the Kalman filter and RTS smoother to obtain Gaussian approximations to  $q_t(x_t)$  and  $q_T(x_t)$ .

There are different ways to form affine approximations to functions, and here we focus on one of those methods—that of using a first-order Taylor series expansion. Assume  $f$  and  $h$  are differentiable, and let  $F_x$  and  $H_x$  denote the Jacobian matrices of  $f$  and  $h$  respectively. Then, at time step  $t$ , we can linearize the dynamic model around the previous (approximate) filtered mean  $m_{t-1}$ :

$$p_t(x_t | x_{t-1}) \approx \mathcal{N}(x_t; A_{t-1}x_{t-1} + b_{t-1}, Q_{t-1}), \quad (2.12)$$

$$A_{t-1} = F_x(m_{t-1}), \quad b_{t-1} = f(m_{t-1}) - F_x(m_{t-1})m_{t-1}.$$

Performing the Kalman prediction step in (2.9) with this approximate affine model yields an approximate predictive distribution  $p_t(x_t | y_{0:t-1}) \approx \mathcal{N}(x_t; m_t^-, P_t^-)$ . Similarly, the measurement model is linearized around  $m_t^-$ :

$$g_t(y_t | x_t) \approx \mathcal{N}(y_t; C_t x_t + d_t, R_t), \quad (2.13)$$

$$C_t = H_x(m_t^-), \quad d_t = h(m_t^-) - H_x(m_t^-)m_t^-,$$

which is used to perform the Kalman update step (2.10). This algorithm is known as the *extended Kalman filter* (EKF, Jazwinski, 1970). Since it is easy to compute gradients in modern programming languages using automatic differentiation libraries (Griewank and Walther, 2008; Elliott, 2018; Bradbury et al., 2018), if the dynamic and measurement models are differentiable, then the EKF is perhaps the simplest Gaussian-approximated filter to implement.

The same affine approximation (2.12) to the dynamic model can be used to perform RTS smoothing and obtain Gaussian approximations to  $q_T(x_t)$ , which is known as the extended RTS smoother (Cox, 1964). In this case, the dynamic model is linearized around the filtered mean  $m_t$ .

### 2.3.1 The iterated extended Kalman smoother

In the extended Kalman filter and RTS smoother, the linearizations were done locally, meaning the dynamic and measurement models were linearized around our current best approximation of the filtered (or smoothed) state. However, once we have performed a smoothing pass, we have improved our knowledge about the true states  $x_{0:T}$ , and can linearize around them and rerun the smoother to obtain a refined estimate. This is the basic idea behind the *iterated extended Kalman smoother* (IEKS, Bell, 1994), which can be seen as a Gauss-Newton optimization algorithm (Nocedal and Wright, 2006, Ch. 10) to find the MAP estimate.

Let  $L(x) = \sum_{p=1}^N r_p(x)^2$  be a nonlinear least-squares objective with residuals  $r_p : X \rightarrow \mathbb{R}$  for  $p \in [1 : N]$ . The Gauss-Newton method is an iterative algorithm to find  $\operatorname{argmin}_x L(x)$ , which approximates  $L(x)$  with a *linear* least-squares objective at each iteration and solves it analytically to obtain the next iterate.

At each iteration  $i$ , the residuals are approximated as affine functions using a first-order Taylor series approximation around the current iterate  $x^{[i-1]}$ .

The problem of finding the MAP estimate of an SSM can be written as

$$x_{0:T}^{\text{MAP}} := \arg \max_{x_{0:T}} q_T(x_{0:T}) = \arg \min_{x_{0:T}} -\log p(x_{0:T} | y_{0:T}),$$

which can be interpreted as minimizing a *loss function*  $L(x_{0:T}) := -\log p(x_{0:T} | y_{0:T})$ . Suppose we have a Gaussian initial condition  $X_0 \sim \mathcal{N}(m_0, P_0)$ , and a nonlinear-Gaussian SSM as specified in (2.11). Then the loss is

$$\begin{aligned} L(x_{0:T}) &= -\log p(x_{0:T} | y_{0:T}) & (2.14) \\ &= \frac{1}{2}(x_0 - m_0)^\top P_0^{-1}(x_0 - m_0) \\ &\quad + \frac{1}{2} \sum_{k=1}^T (x_k - f(x_{k-1}))^\top Q_{k-1}^{-1}(x_k - f(x_{k-1})) \\ &\quad + \frac{1}{2} \sum_{l=0}^T (y_l - h(x_l))^\top R_l^{-1}(y_l - h(x_l)) + \text{constant terms.} \end{aligned}$$

The above expression is a sum of weighted squared residual terms, and we can find a local minimum of  $L(x_{0:T})$  using the Gauss-Newton algorithm.

At iteration  $i$ , we linearize the loss in (2.14) using a first-order Taylor expansion for the dynamic and measurement models around the current guess  $x_{0:T}^{[i-1]}$  (see (2.12) and (2.13)):

$$\begin{aligned} p_t(x_t | x_{t-1}) &\approx \mathcal{N}(x_t; A_{t-1}x_{t-1} + b_{t-1}, Q_{t-1}), \\ g_t(y_t | x_t) &\approx \mathcal{N}(y_t; C_t x_t + d_t, R_t), \end{aligned}$$

where

$$\begin{aligned} A_{t-1} &= F_x(x_{t-1}^{[i-1]}), & b_{t-1} &= f(x_{t-1}^{[i-1]}) - F_x(x_{t-1}^{[i-1]})x_{t-1}^{[i-1]}, \\ C_t &= H_x(x_t^{[i-1]}), & d_t &= h(x_t^{[i-1]}) - H_x(x_t^{[i-1]})x_t^{[i-1]}. \end{aligned}$$

Then, since the mean and mode of a Gaussian distribution are the same, the MAP estimate of the linearized model is the mean trajectory  $m_{0:T}^s$  obtained by running the RTS smoother (Section 2.2.2), so we set  $x_{0:T}^{[i]} = m_{0:T}^s$ . The initial guess  $x_{0:T}^{[0]}$  is typically obtained by running a non-iterated extended RTS smoother and using the mean trajectory.

## 2.4 Feynman–Kac models

We conclude this chapter by introducing a class of sequential models known as *Feynman–Kac* (FK) models (Del Moral, 2004; Chopin and Papaspiliopoulos, 2020). As we will see, Feynman–Kac models provide a flexible formulation that encompasses the filtering and smoothing distributions of SSMs

as special cases. This abstraction is required for modeling some of the sequential decision problems we consider in Chapter 3.

Consider a sequence of random variables  $(X_t)_{t=0}^T$  taking values in a space  $X$  with a joint density

$$\mathbb{M}_T(x_{0:T}) = \mathbb{M}_0(x_0) \prod_{t=1}^T M_t(x_t | x_{0:t-1}),$$

where  $\mathbb{M}_0$  is a prior distribution and  $M_t$  are conditional probability density functions for  $t \in [1 : T]$ . Note that unlike with SSMs, we do not assume that the sequence  $(X_t)_{t=0}^T$  forms a Markov chain. Given so-called *potential functions*  $G_0 : X \rightarrow \mathbb{R}_+$  and  $G_t : X^{t+1} \rightarrow \mathbb{R}_+$  for  $t \in [1 : T]$ , the sequence of distributions defined by

$$\mathbb{Q}_t(x_{0:t}) := \frac{1}{L_t} \mathbb{M}_0(x_0) G_0(x_0) \prod_{s=1}^t M_s(x_s | x_{0:s-1}) G_s(x_{0:s}), \quad t \in [0 : T], \quad (2.15)$$

is called a Feynman-Kac model on  $X^{T+1}$ . The normalizing constant  $L_t$  is

$$L_t = \mathbb{E}_{\mathbb{M}_T} \left[ G_0(X_0) \prod_{s=1}^t G_s(X_{0:s}) \right],$$

and is known as the partition function or the marginal likelihood.

**Example 2.6.** Consider the state-space model defined in (2.3). For a fixed sequence of observations  $y_{0:t}$ , we can trivially associate  $q_t(x_{0:t})$  with a Feynman-Kac model (2.15) by identifying

$$\mathbb{M}_0(x_0) \equiv p_0(x_0), \quad G_0(x_0) \equiv g_0(y_0 | x_0),$$

$$M_t(x_t | x_{0:t-1}) \equiv p_t(x_t | x_{t-1}), \quad G_t(x_{0:t}) \equiv g_t(y_t | x_t), \quad L_t \equiv p(y_{0:t}).$$

Furthermore, the filtering, marginal smoothing and pathwise smoothing distributions are respectively

$$\mathbb{Q}_t(x_t) = p(x_t | y_{0:t}), \quad \mathbb{Q}_T(x_t) = p(x_t | y_{0:T}), \quad \mathbb{Q}_T(x_{0:T}) = p(x_{0:T} | y_{0:T}).$$

By analogy with SSMs, it is straightforward to write recursive equations for the sequence of path-space distributions  $(\mathbb{Q}_t)_{t=0}^T$ . Starting from  $\mathbb{Q}_t$  for some  $t$ , we can compute the *one-step predictive distribution*

$$\mathbb{Q}_t(x_{0:t+1}) = M_{t+1}(x_{t+1} | x_{0:t}) \mathbb{Q}_t(x_{0:t}),$$

and then reweight it with the potential function to obtain  $\mathbb{Q}_{t+1}$ :

$$\mathbb{Q}_{t+1}(x_{0:t+1}) \propto G_{t+1}(x_{0:t+1}) \mathbb{Q}_t(x_{0:t+1}).$$

In Chapter 4, we will introduce methods that make use of these recursive equations to approximate integrals of the form  $\mathbb{E}_{\mathbb{Q}_T}[f(X_{0:T})]$  for suitable functions  $f$ .

### 3. Sequential Decision Making as Inference

In Chapter 1, we alluded to the various connections between decision problems and statistical inference, and in particular how solving a decision problem can be cast as that of inferring optimal decisions. Our goal in this chapter is to formalize this connection, and to specify what we mean in saying "infer optimal decisions". In doing so, we will develop a unified inference and learning framework for sequential decision making under uncertainty, which underpins the contributions of Publications I, II, and IV, as well as the unpublished manuscript Abdulsamad et al. (2023).

Our formulation builds upon the *control-as-inference* paradigm (Toussaint, 2009; Rawlik et al., 2012), while generalizing its scope to encompass a broader class of sequential decision problems, including optimal control, partially observable control, and Bayesian experimental design. The key insight underlying this chapter is that a large variety of sequential decision problems can be expressed as probabilistic inference in suitably defined generative models (Dayan and Hinton, 1997; Kueck et al., 2009; Levine, 2018). This view allows one to treat the selection of optimal controls and the design of informative experiments as instances of a general inference problem: that of inferring the trajectories that are most likely to achieve a desired outcome. By framing decision making as inference, we can draw from the extensive algorithmic toolkit of Bayesian statistics and probabilistic machine learning to solve complex, nonlinear, and non-Gaussian problems.

The chapter is structured as follows. We begin in Section 3.1 with the classical setting of optimal control in Markov decision processes (MDPs), then present the general inference formulation in Section 3.2. Section 3.3 applies this framework to partially observable MDPs, and Section 3.4 demonstrates its application to Bayesian experimental design.

#### 3.1 Stochastic optimal control of Markov decision processes

We consider the problem of a decision maker, or *agent*, acting in an external *environment*. The agent's goal is to select a sequence of decisions or actions

$(A_t)_{t=0}^T$  for  $T \in \mathbb{N}$  to maximize some expected reward. Let  $S \subseteq \mathbb{R}^s$  be the state space representing possible states of the environment and  $A \subseteq \mathbb{R}^a$  be the set of possible actions that can be chosen by the agent. The environment starts in an initial state  $S_0 \sim p_0$ . If the agent takes an action  $a_t$  when the state is  $s_t$ , the environment transitions into a new state  $s_{t+1}$  according to a transition density  $f_t(s_{t+1} | s_t, a_t)$ , and the agent receives an instantaneous reward  $r_t(s_t, a_t) \in \mathbb{R}$ . The terminal reward  $r_T$  is usually defined to only depend on the state. The tuple  $(S, A, p_0, (f_t)_{t=0}^{T-1}, (r_t)_{t=0}^T)$  defines a discrete-time, finite-horizon *Markov decision process* (MDP, Bellman, 1957a; Puterman, 2014).

At each decision time  $t$ , the agent chooses an action according to a time-dependent *stochastic policy*  $\pi_t$ , which specifies a conditional distribution over actions given the current state:  $\pi_t(a_t | s_t)$ . This includes deterministic policies as a special case, where  $\pi_t(\cdot | s_t) = \delta_{g_t(s_t)}(\cdot)$  is a Dirac delta function and  $a_t = g_t(s_t)$  for some function  $g_t$ . Let  $\Pi$  denote the class of admissible policies considered in a given problem. Having the action depend on the current state is known as *feedback* (or *closed-loop*) control; in contrast, an *open-loop* strategy would pre-commit to a sequence  $(a_0, \dots, a_T)$  without conditioning on states.

Given this setup, the stochastic optimal control (SOC) problem is to find a policy that maximizes the *expected cumulative reward*:

$$\max_{\pi \in \Pi} \mathcal{J}(\pi) := \max_{\pi \in \Pi} \mathbb{E}_{p_\pi} \left[ \sum_{t=0}^{T-1} r_t(S_t, A_t) + r_T(S_T) \right], \quad (3.1)$$

where the expectation is under the joint distribution of states and actions,

$$p_\pi(s_{0:T}, a_{0:T}) := p_0(s_0) \pi_0(a_0 | s_0) \prod_{t=1}^T f_t(s_t | s_{t-1}, a_{t-1}) \pi_t(a_t | s_t). \quad (3.2)$$

In addition to enabling adaptive decision making, learning a policy offers the benefit of *amortization*: once learned, it can be deployed for real-time control with negligible computation at each decision point (Amos, 2023).

Throughout this chapter, we assume that the transition densities  $f_t$  and the reward functions  $r_t$  are known. Under this assumption, (3.1) is known alternatively as the *planning* or *policy-optimization* problem in the SOC literature. By contrast, reinforcement learning (RL) typically treats  $(f_t, r_t)$  as unknown and aims to optimize (3.1) from interaction data rather than from a specified model (see, e.g., Sutton and Barto, 2018). We focus on the SOC formulation, which isolates the algorithmic and structural aspects of optimal control without the additional complications of model learning.

### 3.1.1 The dynamic programming solution

The traditional approach to finding the optimal policy is to use *dynamic programming* (DP) algorithms (Bellman, 1957b; Bertsekas, 2012). Observe

that the objective  $\mathcal{J}(\pi)$  in (3.1) can be recursively broken down over time steps to yield *value functions*

$$\begin{aligned} V_T^\pi(s_T) &:= r_T(s_T), \\ V_t^\pi(s_t) &:= \mathbb{E}_{p_\pi} \left[ r_t(S_t, A_t) + V_{t+1}^\pi(S_{t+1}) \mid S_t = s_t \right], \quad t \in [0 : T - 1]. \end{aligned} \quad (3.3)$$

Here  $\mathbb{E}_{p_\pi}[\cdot \mid S_t = s_t]$  denotes conditional expectation given  $S_t = s_t$ . The value function  $V_t^\pi(s_t)$  is the expected cumulative reward attained under the policy  $\pi$  when starting from a state  $s_t$ , and we have  $\mathcal{J}(\pi) = \mathbb{E}_{p_0}[V_0^\pi(S_0)]$ . An optimal policy can be found by recursively solving the following sub-problems for  $t \in [0 : T - 1]$ :

$$V_t^*(s_t) = \max_{\pi_t \in \Pi} \left\{ \mathbb{E}_{p_{\pi_t}} \left[ r_t(S_t, A_t) + V_{t+1}^*(S_{t+1}) \mid S_t = s_t \right] \right\}, \quad (3.4)$$

starting from  $V_T^*(s_T) = r_T(s_T)$ . Equation (3.4) is known as the *Bellman optimality equation* for optimal value functions  $V_t^*$ . Classical DP algorithms like policy and value iteration attempt to solve (3.4) directly, which is only tractable in MDPs with finite state-action spaces, and in continuous MDPs with linear-Gaussian transition functions and quadratic reward functions. For more general settings with arbitrary transition and reward functions, we require approximate solution methods (Sutton and Barto, 2018, Ch. 4).

### 3.1.2 The inference problem

Having outlined DP—and its limitations in high-dimensional or continuous settings—we now adopt a complementary viewpoint: casting SOC as a problem of probabilistic inference. The idea that (3.1) can be reframed as inference has a long history, dating back at least to Sabes and Jordan (1995) and Dayan and Hinton (1997) in single-decision settings, and extended to sequential problems in Attias (2003); Kappen (2005); Toussaint and Storkey (2006); Todorov (2008); Hoffman et al. (2009); Kappen et al. (2012); Rawlik et al. (2012); Levine (2018), among others. Here we follow the formulation of Toussaint (2009) and refer to Rawlik (2013) for a comprehensive overview of related literature and alternatives.

Let  $\tilde{r}_t := \sup_{s_t, a_t} r_t(s_t, a_t)$  be the maximum reward at time  $t$  (with  $r_T(s_T, a_T) := r_T(s_T)$ ), with the assumption that  $\tilde{r}_t < \infty$  for all  $t \in [0 : T]$ . We define *optimality* variables  $(\mathcal{O}_t)_{t=0}^T$  which take values in  $\{0, 1\}$  and have the conditional density

$$p(\mathcal{O}_t = 1 \mid s_t, a_t) = \exp\{\eta r_t(s_t, a_t) - \eta \tilde{r}_t\}, \quad \eta > 0. \quad (3.5)$$

That is, high-reward pairs  $(s_t, a_t)$  are exponentially more likely to be labeled "optimal". Since high-reward trajectories are the only ones of interest, we will henceforth use  $\mathcal{O}_t$  as shorthand for the event  $\{\mathcal{O}_t = 1\}$ . The constant  $\eta$  plays the role of inverse temperature in the Boltzmann distribution: larger  $\eta$  concentrates probability mass on higher-reward choices.

Conditioning state-action trajectories  $(s_{0:T}, a_{0:T})$  on optimality  $\mathcal{O}_{0:T}$  yields the posterior

$$\begin{aligned} q_\pi(s_{0:T}, a_{0:T} | \mathcal{O}_{0:T}) &\propto p_\pi(s_{0:T}, a_{0:T}) \prod_{t=0}^T p(\mathcal{O}_t | s_t, a_t) \\ &\propto p_\pi(s_{0:T}, a_{0:T}) \cdot \exp \left\{ \eta \sum_{t=0}^T r_t(s_t, a_t) \right\}. \end{aligned}$$

This is a state-space model (Section 2.1.1) with  $(s_t, a_t)$  playing the role of the latent "state" and  $\mathcal{O}_t$  playing that of the observation. In contrast to the generative model  $p_\pi$  in (3.2), the posterior  $q_\pi(\cdot | \mathcal{O}_{0:T})$  exponentially favors high-return trajectories. Furthermore, the log marginal likelihood

$$L_T(\pi) := \log p_\pi(\mathcal{O}_{0:T}) = \log \mathbb{E}_{p_\pi} [p(\mathcal{O}_{0:T} | S_{0:T}, A_{0:T})] \quad (3.6)$$

is a measure of the utility of the policy  $\pi$ , since it is the log probability of generating optimal trajectories with  $\pi$ . It is therefore a natural optimization target in the control-as-inference framework, and we seek a maximum likelihood estimate  $\pi^* \in \arg \max_{\pi \in \Pi} \log p_\pi(\mathcal{O}_{0:T})$ .

The MLE differs from the maximizer of the original objective  $\mathcal{J}(\pi)$  in (3.1), since

$$\begin{aligned} \log p_\pi(\mathcal{O}_{0:T}) &= \log \mathbb{E}_{p_\pi} [p(\mathcal{O}_{0:T} | S_{0:T}, A_{0:T})] \\ &\geq \mathbb{E}_{p_\pi} [\log p(\mathcal{O}_{0:T} | S_{0:T}, A_{0:T})] \quad (\text{Jensen's inequality}) \\ &= \eta \mathbb{E}_{p_\pi} \left[ \sum_{t=0}^T r_t(S_t, A_t) \right] + \text{constant}, \end{aligned}$$

where the constant does not depend on  $\pi$ . However, multiplying  $\log p_\pi(\mathcal{O}_{0:T})$  by  $\frac{1}{\eta}$  shows that, up to the additive constant  $-\sum_{t=0}^T \tilde{r}_t$  independent of  $\pi$ , maximizing  $\log p_\pi(\mathcal{O}_{0:T})$  is equivalent to maximizing

$$\frac{1}{\eta} \log p_\pi(\mathcal{O}_{0:T}) = \frac{1}{\eta} \log \mathbb{E}_{p_\pi} \left[ \exp \left\{ \eta \sum_{t=0}^T r_t(S_t, A_t) \right\} \right] =: \mathcal{J}_\eta(\pi).$$

This is a soft maximum (log-sum-exp) of the cumulative reward: it interpolates between the average reward  $\mathcal{J}$  (when  $\eta \rightarrow 0$ ) and the maximum reward achievable under the policy (when  $\eta \rightarrow \infty$ ). It is hence a strict generalization of the original SOC objective  $\mathcal{J}(\pi)$ . For large  $\eta$ , maximizing  $\mathcal{J}_\eta$  favors policies that occasionally achieve exceptional returns over those that optimize average performance. The objective  $\mathcal{J}_\eta$  is known as a risk-sensitive SOC objective (Marcus et al., 1997), and such log-sum-exp versions of standard objectives are used beyond SOC (see, e.g., Li et al., 2023).

**Alternative objectives.** Our focus in this thesis will be on maximizing the log marginal likelihood (3.6), but we wish to highlight that this is far from the only choice in the control-as-inference literature. Here we present one popular alternative. Let  $\bar{\pi}$  be a prior policy and consider the optimality-conditioned posterior

$$p_{\bar{\pi}}(s_{0:T}, a_{0:T} | \mathcal{O}_{0:T}) \propto p_{\bar{\pi}}(s_{0:T}, a_{0:T}) \prod_{t=0}^T p(\mathcal{O}_t | s_t, a_t).$$

For a different candidate policy  $\pi$ , define the trajectory distribution

$$p_{\pi}(s_{0:T}, a_{0:T}) := p_0(s_0) \pi_0(a_0 | s_0) \prod_{t=1}^T f_t(s_t | s_{t-1}, a_{t-1}) \pi_t(a_t | s_t),$$

with which we wish to approximate  $p_{\bar{\pi}}(\cdot | \mathcal{O}_{0:T})$ . Variational inference (Wainwright and Jordan, 2008; Blei et al., 2017) prescribes minimizing the Kullback-Leibler (KL) divergence (Kullback and Leibler, 1951) of  $p_{\pi}$  with respect to the conditioned process  $p_{\bar{\pi}}(\cdot | \mathcal{O}_{0:T})$ :

$$\begin{aligned} \mathbb{D}_{\text{KL}}[p_{\pi} \| p_{\bar{\pi}}(\cdot | \mathcal{O}_{0:T})] &:= \mathbb{E}_{p_{\pi}} \left[ \log \frac{p_{\pi}(S_{0:T}, A_{0:T})}{p_{\bar{\pi}}(S_{0:T}, A_{0:T} | \mathcal{O}_{0:T})} \right] \\ &= \mathbb{E}_{p_{\pi}} \left[ \sum_{t=0}^T \left\{ -\eta r_t(S_t, A_t) + \log \frac{\pi(A_t | S_t)}{\bar{\pi}(A_t | S_t)} \right\} \right] + \text{const.} \end{aligned}$$

The dynamics and initial-state factors cancel in the ratio, leaving only reward and policy terms. If the prior policy  $\bar{\pi}$  is the uniform distribution over the action space, we can ignore the  $\log \bar{\pi}$  term and our optimization problem simplifies to (up to an additive constant)

$$\min_{\pi \in \Pi} \mathbb{D}_{\text{KL}}[p_{\pi} \| p_{\bar{\pi}}(\cdot | \mathcal{O}_{0:T})] = \min_{\pi \in \Pi} \mathbb{E}_{p_{\pi}} \left[ \sum_{t=0}^T \left\{ -\eta r_t(S_t, A_t) + \log \pi(A_t | S_t) \right\} \right]. \quad (3.7)$$

The objective (3.7) is known as the *maximum-entropy RL objective* (Ziebart et al., 2008; Rawlik et al., 2012; Levine, 2018), and underlies many modern deep RL algorithms (see, e.g., Haarnoja et al., 2018).

### 3.1.3 Policy optimization

In Section 3.1.2, we formulated the problem of finding the optimal policy for an MDP as a maximum likelihood estimation problem:

$$\pi^* \in \underset{\pi \in \Pi}{\text{arg max}} \log p_{\pi}(\mathcal{O}_{0:T}),$$

where  $p_{\pi}(\mathcal{O}_{0:T})$  is given in (3.6). We now describe how to find such an optimal policy  $\pi^*$  using a gradient-based optimization procedure.

In practice, we work with a parametric family of policies  $\Pi := \{\pi_{\phi} : \phi \in \Phi\}$ , which reduces the problem to finding  $\phi^* \in \underset{\phi}{\text{arg max}} \log p_{\phi}(\mathcal{O}_{0:T})$ , where  $p_{\phi} :=$

$p_{\pi_\phi}$ . For example, the policy may be represented as a neural network, with  $\phi$  denoting its weights and biases. Now let  $\mathbb{Q}_\phi(s_{0:T}, a_{0:T}) := q_{\pi_\phi}(s_{0:T}, a_{0:T} \mid \mathcal{O}_{0:T})$  be the optimality-conditioned posterior over trajectories under  $\pi_\phi$ . Then the gradient of the log marginal likelihood, also known as the *score*, can be obtained with Fisher’s identity (Cappé et al., 2005):

$$\begin{aligned} S(\phi) &:= \nabla_\phi \log p_\phi(\mathcal{O}_{0:T}) = \mathbb{E}_{\mathbb{Q}_\phi} \left[ \nabla_\phi \log p_\phi(S_{0:T}, A_{0:T}, \mathcal{O}_{0:T}) \right] \\ &= \mathbb{E}_{\mathbb{Q}_\phi} \left[ \sum_{t=0}^T \nabla_\phi \log \pi_\phi(A_t \mid S_t) \right], \end{aligned} \quad (3.8)$$

where the second equality follows because the dynamics and optimality likelihood (3.5) do not depend on  $\phi$ .

The expectation in (3.8) is generally intractable and must be approximated. Assuming for now that we know how to construct a stochastic estimator  $\hat{S}(\phi) \approx \nabla_\phi \log p_\phi(\mathcal{O}_{0:T})$ , we apply stochastic gradient ascent (Robbins and Monro, 1951) to obtain a local maximizer. The high-level procedure is summarized in Algorithm 1.

---

**Algorithm 1:** Maximum likelihood estimation with stochastic gradient ascent.

---

**input** : Initial parameters  $\phi_0$ , step size sequence  $(\beta_i)_{i=1}^\infty$ .

**output** : Local optimum  $\phi^*$  of the marginal likelihood.

```

1  $k \leftarrow 1$ 
2 while not converged do
3   Compute an estimate  $\hat{S}(\phi^{k-1})$  of the score (3.8).
4    $\phi^k \leftarrow \phi^{k-1} + \beta_k \hat{S}(\phi^{k-1})$ 
5    $k \leftarrow k + 1$ 

```

---

How to approximate the score—and hence implement the update—will be our focus in Chapter 4. In the remainder of this chapter, we generalize the learning and inference framework presented for MDPs, and then demonstrate its application to other sequential decision problems.

### 3.2 An inference and learning framework for sequential decision making

The policy learning framework we presented in Section 3.1 can be extended beyond MDPs to other sequential decision problems, yielding algorithms that avoid simplifying assumptions. In this section, we present a general formulation of this approach before specializing it to the specific problems considered in Publications I and II. This general treatment will help clarify the common structure underlying both applications.

Consider a sequence of random variables  $(X_t)_{t=0}^T$  taking values in a space  $\mathcal{X}$ , with joint density  $p_\phi(x_{0:T})$  parameterized by  $\phi \in \Phi$ . We decompose this

density as

$$p_\phi(x_{0:T}) = p_\phi(x_0) \prod_{t=1}^T p_\phi(x_t | x_{0:t-1}).$$

Now, given reward functions  $R_t : X^{t+1} \rightarrow \mathbb{R}$  for  $t \in [0 : T]$ , we wish to solve the optimization problem

$$\max_{\phi} \mathbb{E}_{p_\phi} \left[ \sum_{t=0}^T R_t(X_{0:t}) \right], \quad (3.9)$$

and find parameters  $\phi^*$  which achieve this maximum.

Instead of directly optimizing (3.9), we will consider a more general formulation that offers additional flexibility. Specifically, we optimize the *exponentially tilted* (Li et al., 2023) objective

$$\max_{\phi} \frac{1}{\eta} \log \mathbb{E}_{p_\phi} \left[ \exp \left\{ \eta \sum_{t=0}^T R_t(X_{0:t}) \right\} \right], \quad \eta > 0, \quad (3.10)$$

which recovers the standard objective (3.9) in the limit as  $\eta \rightarrow 0$ . The key insight is that the argument of the logarithm in (3.10) can be recognized as the normalizing constant  $L_T(\phi)$  of the distribution

$$q_\phi(x_{0:T}) := \frac{1}{L_T(\phi)} p_\phi(x_{0:T}) \prod_{t=0}^T \exp\{\eta R_t(x_{0:t})\}. \quad (3.11)$$

Notice that this is an instance of a Feynman–Kac model, which we introduced in Section 2.4. Consequently, the optimization problem (3.10) is equivalent to maximizing the marginal likelihood in the graphical model defined by (3.11). The gradient of the log normalizing constant can be computed using Fisher’s identity (Cappé et al., 2005):

$$\nabla_{\phi} \log L_T(\phi) = \mathbb{E}_{q_\phi} [\nabla_{\phi} \log p_\phi(X_{0:T})],$$

and optimal parameters can be obtained using stochastic gradient ascent as in Algorithm 1.

**Example 3.1** (Control-as-inference for MDPs). *The MDP formulation in Section 3.1 is recovered by defining*

$$\begin{aligned} X_t &:= (S_t, A_t), \\ R_t(X_{0:t}) &= R_t(X_t) := r_t(S_t, A_t), \\ p_\phi(x_t | x_{0:t-1}) &= p_\phi(x_t | x_{t-1}) := f_t(s_t | s_{t-1}, a_{t-1}) \pi_\phi(a_t | s_t). \end{aligned}$$

*This demonstrates that the control-as-inference formulation for MDPs is a special case of the general framework presented here.*

This formulation yields the following general procedure for policy optimization in sequential decision problems:

1. Express the decision-making problem as maximum likelihood estimation in a Feynman-Kac model of the form (3.11).
2. Develop efficient methods to sample from  $q_\phi$  (and thereby estimate the score) using sequential Monte Carlo (Chapter 4).

The power of this general procedure lies in its ability to handle a much broader range of sequential decision problems beyond fully observable MDPs. In the following sections, we demonstrate this flexibility by applying the framework to two important problem classes: partially observable MDPs, where the agent must act under uncertainty about the true state, and sequential experimental design, where the goal is to efficiently gather information about hidden parameters.

### 3.3 Partially observable Markov decision processes

We now apply the general framework from Section 3.2 to the problem of optimal control of *partially observable Markov decision processes* (POMDPs, Åström, 1965; Aoki, 1967; Sondik, 1971), which is the topic of Publication II.

#### 3.3.1 Background

A POMDP extends the Markov decision process framework to settings where the agent cannot directly observe the true state of the system. Instead of observing the state  $S_t$ , at each time step  $t$  the agent receives a noisy observation  $Z_t \in Z$  governed by the conditional density

$$(Z_t | S_t = s_t) \sim g_t(\cdot | s_t).$$

The tuple  $(S, A, Z, p_0, (f_t)_{t=0}^{T-1}, (g_t)_{t=0}^T, (r_t)_{t=0}^T)$  then defines a POMDP, where we have used the same notation for the underlying MDP from Section 3.1. The example of the Apollo lunar module we considered in Chapter 1 is a prototypical example of a POMDP, as the true position and orientation of the spacecraft must be inferred from noisy sensor measurements and used to decide the action (the thrust from the engine).

Since different states may produce identical observations, a single observation does not uniquely identify the current state. However, the full sequence of past observations and actions can provide additional information and constrain the set of plausible states. Optimal decision making must therefore account for the full history of past actions and observations, leading to policies of the form  $\pi_t(a_t | z_{0:t}, a_{0:t-1})$ . An equivalent way to incorporate all past information is to maintain a *belief state*—a probability distribution over the hidden state conditioned on the history,  $p(s_t | z_{0:t}, a_{0:t-1})$ —and make decisions based on this belief. With the above specification for the policy,

the generative process for a POMDP can be written as

$$p_\pi(s_{0:T}, a_{0:T}, z_{0:T}) := p_0(s_0) \prod_{l=1}^T f_l(s_l | s_{l-1}, a_{l-1}) \prod_{t=0}^T \pi_t(a_t | z_{0:t}, a_{0:t-1}) g_t(z_t | s_t).$$

The absence of full observability makes optimal control of POMDPs substantially more challenging than of MDPs. A POMDP can be viewed as an MDP defined over the belief state (Littman et al., 1995), which means the value function now depends on the entire observation-action history:

$$V_t^\pi(z_{0:t}, a_{0:t-1}) = \mathbb{E}_{p_\pi} \left[ r_t(S_t, A_t) + V_{t+1}^\pi(Z_{0:t+1}, A_{0:t}) \mid Z_{0:t} = z_{0:t}, A_{0:t-1} = a_{0:t-1} \right]. \quad (3.12)$$

The expectation is taken over the joint distribution  $p_\pi$  conditioned on the history  $(z_{0:t}, a_{0:t-1})$ . The hidden state  $S_t$  will then be distributed according to the belief state  $p(s_t | z_{0:t}, a_{0:t-1})$ , and the predictive distribution of the next observation  $Z_{t+1}$  given the history is

$$p(z_{t+1} | z_{0:t}, a_{0:t}) = \int g_{t+1}(z_{t+1} | s_{t+1}) f_{t+1}(s_{t+1} | s_t, a_t) p(s_t | z_{0:t}, a_{0:t-1}) ds_{t:t+1}.$$

For finite-horizon POMDPs with discrete state, observation, and action spaces, the value function (3.12) is piecewise linear and convex in the belief space (Sondik, 1971; Smallwood and Sondik, 1973). While this structure allows value iteration to compute the optimal solution in principle, the number of linear segments grows exponentially with both the number of states and the time horizon. Except in very small problems, exact computation quickly becomes infeasible, motivating numerous approximate solution methods (Littman et al., 1995; Kaelbling et al., 1998; Cassandra et al., 1997; Pineau et al., 2003).

On the other hand, for continuous POMDPs, where state, action, and observation spaces are all continuous, the belief state is infinite-dimensional—a probability measure on the continuous state space  $S$ . Consequently, most state-of-the-art methods rely on approximations, some of which impact the agent’s ability to act optimally. For example, many modern deep reinforcement learning algorithms for POMDPs (see, e.g., Hafner et al., 2019; Lee et al., 2020; Zhang et al., 2023) employ the *QMDP approximation* from Littman et al. (1995), which replaces (3.12) with

$$V_t^\pi(z_{0:t}, a_{0:t-1}) \approx \mathbb{E}_{p(\cdot | z_{0:t}, a_{0:t-1})} [V_t^\pi(S_t)],$$

where  $V_t^\pi(S_t)$  is the value function of the underlying fully observable MDP from (3.3). This approximation is equivalent to assuming that all state uncertainty vanishes after the current time step, since we no longer reason about future observations. This assumption is suboptimal because it ignores the information-gathering role of actions. Beyond influencing immediate

rewards, actions can also improve state estimation by directing the agent towards regions of the state space that yield more informative observations, thereby enabling better future decisions. In systems exhibiting this *dual effect*—where actions serve both control and information-gathering purposes—such approximations cannot yield optimal policies (Bar-Shalom and Tse, 1974).

### 3.3.2 The inference formulation

The previous section introduced the problem of optimal control in POMDPs and highlighted the inherent difficulties in continuous settings. We noted that, due to this complexity, most existing algorithms rely on approximations to estimate the value function, often compromising the quality of the learned policy. In this section, we present a novel approach: formulating policy optimization in POMDPs as a maximum likelihood estimation problem, following the general framework of Section 3.2. When combined with the particle methods for score approximation introduced in Chapter 4, this approach yields a policy optimization algorithm that effectively plans in the belief space without requiring simplifying assumptions.

As with MDPs, our objective is to find a policy  $\pi$  that maximizes the expected cumulative reward,

$$\mathcal{J}(\pi) = \mathbb{E}_{p_\pi} \left[ \sum_{t=0}^T r_t(S_t, A_t) \right].$$

However, since the states are not directly observed, it is more natural to work with an equivalent formulation expressed in terms of observations and actions, as given in Proposition 3.2.

**Proposition 3.2** (POMDP objective). *The finite-horizon stochastic optimal control objective for POMDPs can be equivalently written as*

$$\mathcal{J}(\pi) = \mathbb{E}_{p_\pi} \left[ \sum_{t=0}^T R_t(Z_{0:t}, A_{0:t}) \right], \quad (3.13)$$

where  $R_t(z_{0:t}, a_{0:t}) := \mathbb{E}_{p(\cdot | z_{0:t}, a_{0:t-1})} [r_t(S_t, a_t)]$  is the stage reward at time  $t$  associated with the belief state  $p(s_t | z_{0:t}, a_{0:t-1})$  and action  $a_t$ . The expectation is taken with respect to

$$p_\pi(z_{0:T}, a_{0:T}) = \prod_{t=0}^T p(z_t | z_{0:t-1}, a_{0:t-1}) \pi_t(a_t | z_{0:t}, a_{0:t-1}), \quad z_{0:-1} = a_{0:-1} := \emptyset.$$

*Proof.* See Abdulsamad et al. (2025, App. A.1). □

The formulation (3.13) makes explicit the adaptive nature of the decision-making process: at each step  $t$ , the agent observes  $z_t$ , updates its belief

to  $p(s_t | z_{0:t}, a_{0:t-1})$ , selects an action  $a_t$ , and receives an expected reward  $R_t(z_{0:t}, a_{0:t})$ .

By defining  $X_t := (Z_t, A_t)$ , we can recognize that this problem has precisely the structure of the general objective in (3.9). Following the control-as-inference approach, we introduce the optimality-conditioned target distribution

$$q_\pi(z_{0:T}, a_{0:T}) \propto p_\pi(z_{0:T}, a_{0:T}) \prod_{t=0}^T \exp\{\eta R_t(z_{0:t}, a_{0:t})\}, \quad \eta > 0. \quad (3.14)$$

If we can generate samples from  $q_\pi$ , then we can approximate the score and apply the maximum likelihood estimation procedure from Algorithm 1 to optimize the policy. However, sampling from  $q_\pi$  is non-trivial: evaluating the reward  $R_t(z_{0:t}, a_{0:t})$  requires marginalizing over the belief state  $p(s_t | z_{0:t}, a_{0:t-1})$ , while the trajectory distribution  $p_\pi(z_{0:T}, a_{0:T})$  couples observations through these evolving belief states. Simulating an MDP over beliefs cannot typically be done analytically, which makes exact sampling infeasible and requires specialized approximate inference techniques.

**Remark 3.3.** *The key algorithmic challenge is sampling from  $q_\pi$ , the distribution over observation-action trajectories conditioned on optimality. In Publication II, we develop a nested sequential Monte Carlo (SMC) algorithm for this purpose. The outer SMC loop simulates trajectories  $(z_{0:t}^n, a_{0:t}^n)$  forward in time, while an inner SMC algorithm maintains an empirical approximation of the belief state  $p(s_t | z_{0:t}^n, a_{0:t-1}^n)$  for each observation-action trajectory.*

### 3.4 Bayesian experimental design

As a final example of the generality of the learning framework developed in Section 3.2, we now consider a different type of decision problem: that of optimally designing experiments (Fisher, 1935; Huan et al., 2024). We adopt a Bayesian decision-theoretic perspective which leads to a framework known as *Bayesian experimental design* (BED, Lindley, 1956; Bernardo and Smith, 1994), which forms the basis of Publication I.

#### 3.4.1 Background

Consider an experimenter who, upon choosing an action (or *design*)  $\Xi = \xi$ , observes an outcome  $Y = y$  with probability  $f_\Theta(y | \xi)$ , a family indexed by an unknown (random) parameter  $\Theta$ . The experimenter has a preference for certain experimental outcomes over others, captured in a *utility function*  $u(\xi, y, \theta)$  which quantifies how useful they find the experiment  $(\xi, y)$  if the true parameter is  $\Theta = \theta$ . The experimenter's goal is to choose a design to

execute, and the rational choice is to apply that design which maximizes the *expected utility* (von Neumann and Morgenstern, 1944)

$$U(\xi) := \mathbb{E}_{p(\cdot|\xi)}[u(\xi, Y, \Theta)], \quad (3.15)$$

where the expectation is taken under  $p(y, \theta | \xi) = f_\theta(y | \xi)p(\theta)$ , and  $p(\theta)$  is a prior distribution encoding the experimenter's beliefs about  $\Theta$  prior to performing the experiment.

Little more can be said without specifying a utility function, which depends on the problem. Nevertheless, the experimenter is often either unclear on how to define an appropriate utility function, or is primarily interested in *inferring*  $\Theta$  (or some function of it), as illustrated in the example below.

**Example 3.4.** *A pharmaceutical company wants to determine the price at which to market a new drug. One key factor influencing this decision is the optimal dosage  $\Theta$ . The scientists in charge of clinical trials must decide which dosage levels to test and how to allocate participants across them—the design  $\Xi$ —for a fixed trial size. Their goal is to estimate  $\Theta$  as precisely as possible, leaving it to the executives to use this information, along with other considerations, to set the price.*

Even for this special case of inferring  $\Theta$ , many different utility functions have been proposed over the years (see, e.g., Chaloner and Verdinelli, 1995). We will focus on the proposal of Lindley (1956) of using the *information gain* in  $\Theta$  from the prior to the posterior as the utility function:

$$u(\xi, y, \theta) := \log p(\theta | y, \xi) - \log p(\theta),$$

based on Shannon's definition of information as negative log probability (Shannon, 1948). The interpretation is intuitive: the experimenter prefers experiments  $(\xi, y)$  which make the true parameter value  $\theta$  more probable under the posterior. Substituting this expression into (3.15) yields a decision criterion known as the *expected information gain* (EIG),

$$\begin{aligned} U(\xi) &= \mathbb{E}_{p(\cdot|\xi)}[\log p(\Theta | Y, \xi) - \log p(\Theta)] \\ &= \mathbb{H}[\Theta] - \mathbb{H}[\Theta | Y, \xi], \end{aligned} \quad (3.16)$$

where the two terms in (3.16) denote the entropy and conditional entropy, respectively:

$$\mathbb{H}[\Theta] := - \int \log p(\theta) p(\theta) d\theta, \quad \mathbb{H}[\Theta | Y, \xi] := - \int \log p(\theta | y, \xi) p(y, \theta | \xi) d\theta dy.$$

The EIG is hence the expected reduction in entropy from prior to posterior, or equivalently, the *mutual information* (MacKay, 2003) between  $\Theta$  and  $Y$  for a fixed design  $\xi$ .

The expected information gain is both conceptually appealing and decision-theoretically justified (Bernardo, 1979), providing a principled criterion for experimental design. Unfortunately, optimizing the EIG in practice is notoriously difficult because it involves nested expectations over unknown quantities. In particular, the posterior  $p(\theta | y, \xi)$  required for computing the conditional entropy is rarely available in closed form, and even when it is, the requisite integrals may lack analytical expressions. As a result, optimizing the EIG typically requires approximations or sampling-based methods, even for relatively simple models.

### 3.4.2 Designing sequential experiments

In the previous section, we focused on designing a single experiment. A more interesting and widely applicable scenario is when we wish to optimize a *sequence of experiments*, where the outcomes of previous experiments are used to select the design for the next. For instance, recall the example of the psychologist studying temporal discounting from Chapter 1. They ask a participant a question, update their beliefs based on the response, then decide on and ask the next question. This process repeats until they are sufficiently confident about the participant’s true discount rate. We consider this setting of sequential experiments now.

Let  $T$  be the total number of experiments to conduct. At each  $t \in [1 : T]$ , the experimenter selects a design  $\xi_t$  and observes an outcome  $Y_t$ . In the most general case, the distribution of  $Y_t$  may depend on all previous observations  $Y_{1:t-1}$ , all previous designs including the current one  $\xi_{1:t}$ , and the hidden parameter  $\Theta$ . We denote this conditional density by  $f_{\Theta}(y_t | y_{1:t-1}, \xi_{1:t})$ , which may also vary with  $t$ .

Extending the definition of the expected information gain to a sequence of experiments is straightforward. If the full sequence of designs  $\xi_{1:T}$  is fixed in advance, the expected utility is

$$U(\xi_{1:T}) = \mathbb{H}[\Theta] - \mathbb{H}[\Theta | Y_{1:T}, \xi_{1:T}],$$

which is the expected reduction in uncertainty in  $\Theta$  after all experiments have been conducted.

In a more flexible setting, the experimenter adapts each new design to the data observed so far. We formalize this adaptive behavior through a stochastic policy  $\pi$  which specifies a conditional distribution over designs given an experimental history,  $\pi(\xi_t | y_{1:t-1}, \xi_{1:t-1})$ . This enables the experimenter to choose better designs and, when the policy is learned prior to performing experiments, confers the usual amortization benefit: experiments can be run in real time because the experimenter queries the policy rather than solving a new optimization problem at every time step (Rainforth et al., 2024).

Under such a policy, the sequence of designs  $\Xi_{1:T}$  is random, and the

expected information gain becomes a function of the policy,

$$U(\pi) = \mathbb{H}[\Theta] - \mathbb{H}[\Theta \mid Y_{1:T}, \Xi_{1:T}]. \quad (3.17)$$

The expectation for the conditional entropy is taken under the joint distribution

$$p_\pi(y_{1:T}, \xi_{1:T}) := \prod_{t=1}^T \pi(\xi_t \mid y_{1:t-1}, \xi_{1:t-1}) p(y_t \mid y_{1:t-1}, \xi_{1:t}), \quad (3.18)$$

$$p(y_t \mid y_{1:t-1}, \xi_{1:t}) = \int f_\theta(y_t \mid y_{1:t-1}, \xi_{1:t}) p(d\theta \mid y_{1:t-1}, \xi_{1:t-1}),$$

with the convention  $y_{1:0} = \xi_{1:0} := \emptyset$ .

Most existing works on learning policies for Bayesian experiment design (e.g., Huan and Marzouk, 2016; Foster et al., 2021; Ivanova et al., 2021; Blau et al., 2022) optimize tractable approximations or bounds on the expected information gain (Foster et al., 2019; Rainforth et al., 2024), which is often intractable to evaluate, as noted above.

### 3.4.3 The inference formulation

Our goal in this section is to show that, although a different problem, Bayesian experimental design can be cast in terms similar to the optimal control of POMDPs (Section 3.3). Consequently, we can use the inference and learning procedure from Section 3.2 to learn an optimal policy. Define  $X_t := (\Xi_t, Y_t)$ , and consider the following alternative expression for the EIG in (3.17).

**Proposition 3.5** (EIG as expected cumulative reward). *The expected information gain for the sequential problem (3.17) can be expressed as*

$$U(\pi) = \mathbb{E}_{p_\pi} \left[ \sum_{t=1}^T R_t(X_{1:t}) \right], \quad (3.19)$$

where the expectation is under (3.18) and  $R_t(x_{1:t}) = \alpha_t(x_{1:t}) + \beta_t(x_{1:t})$  is a stage reward with components  $\alpha_t$  and  $\beta_t$  defined as

$$\alpha_t(x_{1:t}) = \int \log f_\theta(y_t \mid x_{1:t-1}, \xi_t) p(\theta \mid x_{1:t}) d\theta, \quad (3.20)$$

$$\beta_t(x_{1:t}) = -\log \int f_\theta(y_t \mid x_{1:t-1}, \xi_t) p(\theta \mid x_{1:t-1}) d\theta.$$

*Proof.* See Iqbal et al. (2024, App. A). □

The components of the stage reward  $R_t$  in (3.20) have simple interpretations.  $\beta_t(x_{1:t})$  is the *surprisal* or informativeness (Shannon, 1948) of the experiment  $x_t$  under the current belief  $p(\theta \mid x_{1:t-1})$ , while  $\alpha_t(x_{1:t})$  represents the expected log likelihood of the same observation under the *updated*

posterior distribution  $p(\theta | x_{1:t})$ , capturing how well the posterior model accounts for the new data. In essence,  $\beta_t$  quantifies predictive surprise and  $\alpha_t$  rewards posterior consistency, together decomposing the incremental information gain into these two complementary aspects.

Observe that the EIG (3.19) is an expected cumulative reward of the form (3.9), and hence the learning formulation of Section 3.2 is applicable directly. Specifically, the normalizing constant of the distribution

$$q_\pi(x_{1:T}) \propto p_\pi(x_{1:T}) \prod_{t=1}^T \exp\{\eta R_t(x_{1:t})\}, \quad \eta > 0, \quad (3.21)$$

where  $p_\pi$  is defined in (3.18), is a tilted version of the objective (3.19). From Section 3.2, we know that if we have a sampler for  $q_\pi$ , we can perform stochastic gradient ascent to optimize  $\pi$ .

Similar to the case for POMDPs (Section 3.3.2), sampling from  $q_\pi$  in (3.21) is not straightforward. Computing  $R_t$  (3.20) requires integrating with respect to the filtering distributions  $p(\theta | x_{1:t})$ . Moreover, the prior distribution over experiments under the policy,  $p_\pi$  in (3.18), also involves expectations over the parameter posteriors. Taken together, these requirements preclude exact sampling from  $q_\pi$  and necessitate developing approximate sampling schemes.

**Remark 3.6.** *In Publication I, we introduce a sequential Monte Carlo method called Inside-Out SMC<sup>2</sup> (IO-SMC<sup>2</sup>) to sample from  $q_\pi$  (3.21). IO-SMC<sup>2</sup> uses an algorithm known as iterated batch importance sampling (IBIS, Chopin, 2002) to track the parameter posteriors, which operates inside another SMC algorithm that simulates experiments  $(\Xi_{1:T}, Y_{1:T})$ . IBIS is discussed in detail in Section 4.3.2. In Publication IV, we introduce a less exact but computationally lighter variant of IO-SMC<sup>2</sup>, the Inside-Out Nested Particle Filter (IO-NPF).*



## 4. Particle Approximations in General Sequential Models

In Chapter 3, we formulated several sequential decision problems as maximum likelihood estimation in appropriate Feynman–Kac models, and presented a general gradient-based algorithm in Section 3.2 to find an MLE. This procedure is contingent on our ability to estimate the score, which is an expectation of the form  $\mathbb{E}_{q_\phi}[f(X_{0:T})]$  where

$$q_\phi(x_{0:T}) \propto p_\phi(x_{0:T}) \prod_{t=0}^T \exp\{\eta R_t(x_{0:t})\}.$$

The question now is how to compute this expectation, because it is intractable in general.

Chapter 2 introduced various recursive algorithms that form Gaussian approximations for state-space models. While they are broadly used, they are insufficient for our present purposes for two key reasons. First, the Gaussian methods we covered assume Markovian state dynamics, an assumption that no longer holds for many decision problems (Sections 3.3.2 and 3.4.3). Second, Gaussian approximations are fundamentally limited in their ability to capture distributions that exhibit multimodality, heavy tails, or other characteristics that deviate substantially from quadratic log-densities.

To overcome these limitations, this chapter develops a more flexible class of approximations that represent distributions as finite collections of weighted sample points. This representation enables us to approximate the required integral as

$$\mathbb{E}_{q_\phi}[f(X_{0:T})] \approx \sum_{n=1}^N W^n f(X_{0:T}^n),$$

where the weights  $W^n \in \mathbb{R}_+$  and the trajectories  $X_{0:T}^n \in \mathcal{X}^{T+1}$  for  $n \in [1:N]$ , with  $N \in \mathbb{N}$  denoting the number of particles.

**Notation and conventions.** Throughout this chapter, we adopt measure-theoretic notation to easily represent both continuous distributions and empirical distributions defined by Dirac measures. For a measure  $\mu$ , we

write  $\mu(dx)$  to denote the measure of an infinitesimal region around  $x \in X$ , emphasizing the variable being sampled or integrated over. The following expressions for expectations are equivalent:

$$\mu(f) := \mathbb{E}_\mu[f(X)] = \int_X f(x) d\mu(x) = \int_X f(x) \mu(dx).$$

When the measure  $\mu$  is absolutely continuous with respect to the Lebesgue measure, we use the same symbol for its density, so that  $\mu(dx) = \mu(x)dx$ . This notation follows that in standard textbooks on sequential Monte Carlo (see, e.g., Chopin and Papaspiliopoulos, 2020) and simplifies the presentation considerably.

Now we can formally state the main goal of this chapter: constructing particle approximations to arbitrary distributions  $\nu$ . A *particle approximation* is an empirical measure of the form

$$\nu^N(dx) := \sum_{n=1}^N W^n \delta_{X^n}(dx), \quad (4.1)$$

where  $N \in \mathbb{N}$ ,  $X^{1:N} = (X^1, \dots, X^N)$  are  $X$ -valued random variables, and  $W^{1:N} = (W^1, \dots, W^N)$  are non-negative random weights satisfying  $\sum_{n=1}^N W^n = 1$  almost surely. Here,  $\delta_{X^n}$  is the Dirac measure concentrated at  $X^n$ .

We will use the term "particle approximation" to refer both to the empirical measure  $\nu^N$  in (4.1) and to its weighted support set  $\{(X^n, W^n)\}_{n=1}^N$ . These approximations are useful for two main purposes:

- *Sampling*: A sample  $X \sim \nu^N$  can be collected by first sampling an index  $A$  from the categorical distribution over  $N$ , with the probability of sampling  $n \in [1:N]$  given by  $W^n$ , and setting  $X = X^A$ .
- *Integration*: For suitable functions  $f$ , we can approximate the integral

$$\nu(f) = \int_X f(x) \nu(dx)$$

by the weighted sum  $\nu^N(f) = \sum_{n=1}^N W^n f(X^n)$ .

The rest of this chapter is organized in two parts. First, we study particle approximations for *static distributions*, where the goal is to approximate a fixed target distribution  $\nu$ —covering classical Monte Carlo and its extension, importance sampling. Second, we consider sequential models, in which the target distributions are specified as Feynman–Kac models (Section 2.4). We discuss how we can construct particle approximations for FK models recursively, yielding the family of methods collectively known as *sequential Monte Carlo*.

## 4.1 The static setting

We begin with the case of static target distributions, where the goal is to form particle approximations of a fixed probability measure  $\nu$ . Throughout the section, we use  $f : X \rightarrow \mathbb{R}$  to denote an arbitrary function.

### 4.1.1 The Monte Carlo method

The classical Monte Carlo method (Metropolis and Ulam, 1949; Roger, 1987) provides the simplest example of a particle approximation. It draws samples  $X^{1:N}$  independently from  $\nu$  and defines the empirical measure

$$\nu^N(\mathrm{d}x) := \frac{1}{N} \sum_{n=1}^N \delta_{X^n}(\mathrm{d}x). \quad (4.2)$$

The validity of (4.2) as an approximation to  $\nu$  can be justified in different ways. For any function  $f$  such that  $\nu(f) < \infty$ ,  $\nu^N(f)$  is a consistent estimator of  $\nu(f)$  by Kolmogorov's strong law of large numbers (see, e.g., Kallenberg, 2021, Theorem 5.23):

$$\nu^N(f) = \frac{1}{N} \sum_{n=1}^N f(X^n) \rightarrow \nu(f) \quad \text{almost surely as } N \rightarrow \infty.$$

Moreover, the Monte Carlo estimator is unbiased,  $\mathbb{E}[\nu^N(f)] = \nu(f)$ , and its *mean squared error* (MSE) is

$$\mathbb{E} \left[ (\nu^N(f) - \nu(f))^2 \right] = \frac{1}{N} \mathrm{Var}_\nu[f(X)],$$

where  $\mathrm{Var}_\nu[f(X)]$  denotes the variance of  $f$ . Hence, when  $\mathrm{Var}_\nu[f(X)] < \infty$ , the MSE decreases at rate  $\mathcal{O}(1/N)$ .

### 4.1.2 Importance sampling

The Monte Carlo method requires samples from the target distribution  $\nu$ , which in many practical scenarios may not be readily available. A classic example is when we want to compute an expectation under a Bayesian posterior:

$$p(\mathrm{d}x | y) = \frac{p(y | x)p(\mathrm{d}x)}{p(y)}.$$

Computing the marginal likelihood  $p(y) = \int p(y | x)p(\mathrm{d}x)$  is often intractable, precluding direct sampling from  $\nu$ .

Suppose we know another distribution  $\mu$  (which we will call the *proposal*) that is easy to sample from, such that  $\mu(x) > 0$  whenever  $\nu(x) > 0$ . *Importance sampling* constructs a particle approximation for  $\nu$  using samples from  $\mu$ . Assume we can evaluate the ratio

$$w(x) \propto \frac{\nu(x)}{\mu(x)}, \quad x \in X,$$

in closed form, up to a multiplicative constant. Then, for any appropriate function  $f$ , we have the identity

$$v(f) = \mu \left( f \frac{v}{\mu} \right) = \frac{\mu(wf)}{\mu(w)},$$

and we can form a Monte Carlo estimate of the RHS by sampling from  $\mu$ :

$$v(f) \approx \frac{\sum_{n=1}^N w(X^n) f(X^n)}{\sum_{m=1}^N w(X^m)}, \quad X^n \sim \mu.$$

This corresponds to the particle approximation

$$v^N(dx) := \sum_{n=1}^N W^n \delta_{X^n}(dx), \quad W^n := \frac{w(X^n)}{\sum_{m=1}^N w(X^m)},$$

where  $W^n$  are called *importance weights*.  $v^N$  depends on  $\mu$ , but we omit this dependence from the notation for conciseness. Unlike standard Monte Carlo, the estimate  $v^N(f)$  is biased because it is a ratio of unbiased estimators, but it is still consistent (Owen, 2013, Ch. 9).

### 4.1.3 The effective sample size

We conclude this section with a short discussion on how to judge the quality of an importance sampling approximation in practice. The most popular method among practitioners is to compute what is known as the *effective sample size* (ESS, Liu and Chen, 1995; Liu, 1996):

$$\text{ESS}(W^{1:N}) := \frac{1}{\sum_{n=1}^N (W^n)^2}.$$

The ESS is an intuitive measure of the diversity of a particle approximation: it obtains its maximum value of  $N$  when all weights are equal to  $1/N$  and its minimum value of 1 when exactly one weight is nonzero. A low value for the ESS indicates that the proposal is poorly suited for the target. For further insights into measuring and improving the quality of importance sampling approximations, see Agapiou et al. (2017); Chatterjee and Diaconis (2018).

## 4.2 The sequential setting

We now turn our attention to particle approximations for sequential models, focusing on the Feynman–Kac (FK) formulation from Section 2.4. As discussed in Sections 3.3.2 and 3.4.3, we can frame certain sequential decision problems as maximum likelihood estimation in FK models, sampling from which is necessary to compute the score. We begin with a direct extension of importance sampling to time-evolving targets and then introduce more effective procedures that address its well-known limitations, leading to the class of *sequential Monte Carlo* methods.

### 4.2.1 Sequential importance sampling

Our goal is to form particle approximations to the FK model  $\mathbb{Q}_t$  from (2.15),

$$\mathbb{Q}_t(\mathrm{d}x_{0:t}) := \frac{1}{L_t} \mathbb{M}_0(\mathrm{d}x_0) G_0(x_0) \prod_{s=1}^t M_s(\mathrm{d}x_s | x_{0:s-1}) G_s(x_{0:s}), \quad t \in [0:T],$$

so that we can estimate  $\mathbb{E}_{\mathbb{Q}_t}[f(X_{0:t})]$  for suitable functions  $f$ . A direct approach is importance sampling on  $X^{t+1}$  (Section 4.1.2). In particular, if we construct a proposal that factorizes over time:

$$\mathbb{P}_t(\mathrm{d}x_{0:t}) = \mathbb{P}_0(\mathrm{d}x_0) \prod_{s=1}^t P_s(\mathrm{d}x_s | x_{0:s-1}), \quad (4.3)$$

then the unnormalized importance weight at time  $t$  factorizes as

$$\begin{aligned} w_t^n &:= w_t(X_{0:t}^n) \propto \frac{\mathbb{Q}_t(X_{0:t}^n)}{\mathbb{P}_t(X_{0:t}^n)} \\ &\propto \frac{\mathbb{Q}_{t-1}(X_{0:t-1}^n) \mathbb{Q}_t(X_t^n | X_{0:t-1}^n)}{\mathbb{P}_{t-1}(X_{0:t-1}^n) P_t(X_t^n | X_{0:t-1}^n)} \\ &\propto w_{t-1}(X_{0:t-1}^n) \cdot \frac{M_t(X_t^n | X_{0:t-1}^n) G_t(X_{0:t}^n)}{P_t(X_t^n | X_{0:t-1}^n)} =: w_{t-1}^n \alpha_t(X_{0:t}^n), \end{aligned} \quad (4.4)$$

where  $w_{t-1}$  is the unnormalized importance weight function at time  $t-1$  and  $\alpha_t$  is called the incremental weight function.

The factorization (4.4) yields a simple recursive scheme. If we have an importance sampling approximation  $\mathbb{Q}_{t-1}^N(\mathrm{d}x_{0:t-1}) = \sum_{n=1}^N W_{t-1}^n \delta_{X_{0:t-1}^n}(\mathrm{d}x_{0:t-1})$  of  $\mathbb{Q}_{t-1}$ , we can approximate  $\mathbb{Q}_t$  by sampling  $X_{0:t}^n \sim P_t(\mathrm{d}x_t | x_{0:t-1}) \delta_{X_{0:t-1}^n}(\mathrm{d}x_{0:t-1})$  and computing importance weights using (4.4) to yield

$$\mathbb{Q}_t^N(\mathrm{d}x_{0:t}) = \sum_{n=1}^N W_t^n \delta_{X_{0:t}^n}(\mathrm{d}x_{0:t}), \quad W_t^n = \frac{w_t^n}{\sum_{m=1}^N w_t^m}.$$

This procedure, known as *sequential importance sampling* (SIS), is consistent ( $\mathbb{Q}_t^N(f) \rightarrow \mathbb{Q}_t(f)$  almost surely as  $N \rightarrow \infty$ ), and  $\sum_{n=1}^N W_{t-1}^n \alpha_t(X_{0:t}^n)$  is an unbiased estimator of the normalizing constant increment  $L_t/L_{t-1}$  (see, e.g., Doucet and Johansen, 2011).

While simple to implement and understand, SIS suffers from a problem known as *weight degeneracy*: the weights accumulate on a few trajectories as  $t$  grows, causing the effective sample size to collapse and the estimator's variance to increase (often rapidly, see Chopin and Papaspiliopoulos, 2020, Sec. 8.7). For this reason, plain SIS is rarely used in practice, and we now describe a strategy to partially alleviate degeneracy.

### 4.2.2 Resampling

*Resampling* refers to the technique of obtaining an unweighted particle set by repeatedly sampling from a set of weighted particles. For example, in

the *multinomial resampling scheme*, given a particle approximation

$$\mathbb{Q}_{t-1}^N(\mathrm{d}\mathbf{x}_{t-1}) = \sum_{n=1}^N W_{t-1}^n \delta_{X_{t-1}^n}(\mathrm{d}\mathbf{x}_{t-1}),$$

we draw indices  $A_t^n \sim \mathcal{C}(W_{t-1}^{1:N})$  from the categorical distribution  $\mathcal{C}(W_{t-1}^{1:N})$ , which generates value  $n$  with probability  $W_{t-1}^n$ , and form the measure

$$\tilde{\mathbb{Q}}_{t-1}^N(\mathrm{d}\mathbf{x}_{t-1}) = \frac{1}{N} \sum_{n=1}^N \delta_{X_{t-1}^{A_t^n}}(\mathrm{d}\mathbf{x}_{t-1}).$$

Observe that resampling only hides degeneracy: even though we now have equal weights, we have not generated new particles, and we will have duplicates. Furthermore, it is obvious that the estimate  $\tilde{\mathbb{Q}}_{t-1}^N(f)$  has higher variance than  $\mathbb{Q}_{t-1}^N(f)$ , because of randomly sampling the ancestor indices. How then is resampling useful?

The utility of resampling lies in its ability to counteract the exponential growth of the variance of the importance weights caused by repeated weight multiplication in subsequent steps. By discarding low-weight particles and replicating high-weight ones, resampling prevents weight skewness from rendering the approximation useless over time (Chopin and Papaspiliopoulos, 2020; Doucet et al., 2001). Although this introduces additional noise at the current step, it stabilizes the estimator in the long run, and there is considerable empirical evidence that resampling improves the performance of the SIS (Chopin et al., 2022).

Multinomial resampling has relatively high variance—notice that it would discard particles even when  $\mathbb{Q}_{t-1}^N$  has uniform weights. In practice, lower-variance schemes such as *systematic*, *stratified*, or *residual* resampling are preferred (Carpenter et al., 1999; Douc et al., 2005).

### 4.2.3 Sequential Monte Carlo

We now present the general *sequential Monte Carlo* (SMC) algorithm, also known as the *particle filter*. It combines sequential importance sampling with resampling to reduce degeneracy, see Algorithm 2. The original particle filter introduced in Gordon et al. (1993) used  $\mathbb{P}_t = \mathbb{M}_t$ , a choice that has come to be known as the *bootstrap*.

Various results on convergence and consistency of the particle approximations, and on unbiasedness of the normalizing constant estimates, are available in Crisan (2001); Crisan and Doucet (2002); Del Moral (2004); Chopin and Papaspiliopoulos (2020).

In Algorithm 2, the particles are resampled at every time step, which is usually unnecessary and can reduce particle diversity. It suffices to resample only when the variance of the weights is high, which is typically measured using the effective sample size (Section 4.1.2). This is known

**Algorithm 2:** Sequential Monte Carlo.

---

**input** : A Feynman–Kac model  $(\mathbb{Q}_t)_{t=0}^T$  (2.15), an importance proposal  $\mathbb{P}_T$  (4.3) and number of particles  $N$ .

**output** : Particle approximation  $\mathbb{Q}_t^N(\mathrm{d}x_{0:t}) = \sum_{n=1}^N W_t^n \delta_{X_{0:t}^n}(\mathrm{d}x_{0:t})$  and normalizing constant estimate  $L_t^N$  for  $t \in [0 : T]$ .

- 1  $\triangleright$  Operations involving the index  $n$  must be performed for  $n \in [1 : N]$ .
- 2 Sample  $X_0^n \sim \mathbb{P}_0(\mathrm{d}x_0)$
- 3 Set  $w_0^n \leftarrow \frac{\mathbb{M}_0(X_0^n)G_0(X_0^n)}{\mathbb{P}_0(X_0^n)}$  and  $W_0^n \leftarrow w_0^n / \sum_{m=1}^N w_0^m$
- 4 Set  $L_0^N \leftarrow \frac{1}{N} \sum_{m=1}^N w_0^m$
- 5 **for**  $t \leftarrow 1$  **to**  $T$  **do**
- 6     Sample  $A_t^n \sim \mathcal{C}(W_{t-1}^{1:N})$
- 7     Sample  $X_{0:t}^n \sim P_t(\mathrm{d}x_t | x_{0:t-1}) \delta_{X_{0:t-1}^{A_t^n}}(\mathrm{d}x_{0:t-1})$
- 8     Set  $w_t^n \leftarrow \frac{M_t(X_t^n | X_{0:t-1}^n)G_t(X_{0:t}^n)}{P_t(X_t^n | X_{0:t-1}^n)}$  and  $W_t^n \leftarrow w_t^n / \sum_{m=1}^N w_t^m$
- 9     Set  $L_t^N \leftarrow L_{t-1}^N \cdot \frac{1}{N} \sum_{n=1}^N w_t^n$

---

as the *adaptive* resampling strategy, when resampling is only performed if  $\text{ESS}(W_t^{1:N}) < \beta N$  for some  $\beta \in (0, 1)$  (Liu and Chen, 1995; Del Moral et al., 2012).

#### 4.2.4 Sequential Monte Carlo Smoother

The SMC algorithm of the previous section produces particle approximations to the (pathwise) filtering distributions  $\mathbb{Q}_t(\mathrm{d}x_{0:t})$ . It also yields an approximation of the pathwise smoothing distribution,  $\mathbb{Q}_T^N(\mathrm{d}x_{0:T}) = \sum_{n=1}^N W_T^n \delta_{X_{0:T}^n}(\mathrm{d}x_{0:T})$ , from which the marginals can be extracted. However, using this forward pass alone to obtain the marginal smoothing distributions is often insufficient due to the problem of *sample impoverishment*: repeated resampling collapses lineages of particles so that only a few distinct ancestors survive (Kitagawa, 1996).

Various algorithms have been proposed to address sample impoverishment (see, e.g., Briers et al., 2010; Douc et al., 2011; Lindsten and Schön, 2013). Here, we present the *backward sampling* algorithm from Godsill et al. (2004), which generates diverse trajectories by sampling backward with approximate reverse kernels  $\mathbb{Q}_T(\mathrm{d}x_{0:t} | x_{t+1:T})$ . Using the identity

$$\mathbb{Q}_T(\mathrm{d}x_{0:t} | x_{t+1:T}) \propto \mathbb{Q}_T(\mathrm{d}x_{0:t}, x_{t+1:T}) = \frac{\mathbb{Q}_T(x_{0:T})}{\mathbb{Q}_t(x_{0:t})} \mathbb{Q}_t(\mathrm{d}x_{0:t})$$

and the particle approximation  $\mathbb{Q}_t^N(\mathrm{d}x_{0:t}) = \sum_{n=1}^N W_t^n \delta_{X_{0:t}^n}(\mathrm{d}x_{0:t})$  obtained via Algorithm 2, we obtain the empirical distribution

$$\bar{\mathbb{Q}}_T^N(\mathrm{d}x_{0:t} | x_{t+1:T}) = \sum_{n=1}^N \bar{W}_t^n \delta_{X_{0:t}^n}(\mathrm{d}x_{0:t}),$$

where the smoothing weights  $\bar{W}_t^n$  are given by

$$\bar{W}_t^n := \frac{\bar{w}_t^n}{\sum_{m=1}^N \bar{w}_t^m}, \quad \bar{w}_t^n := \frac{\mathbb{Q}_T(X_{0:t}^n, x_{t+1:T})}{\mathbb{Q}_t(X_{0:t}^n)} W_t^n. \quad (4.5)$$

The backward-sampling procedure is: (i) draw a terminal index  $A_T \sim \mathcal{C}(W_T^{1:N})$  and set  $\bar{X}_T = X_T^{A_T}$ ; (ii) for  $t \in \{T-1, \dots, 0\}$ , sample an ancestor  $A_t \sim \mathcal{C}(\bar{W}_t^{1:N})$  and set  $\bar{X}_{t:T} = (X_t^{A_t}, \bar{X}_{t+1:T})$ . Repeating this yields a collection of trajectories approximately distributed according to the smoothing law with markedly improved path diversity.

The cost of sampling  $N$  trajectories with backward sampling scales as  $\mathcal{O}(N^2)$ , because we need to compute the smoothing weights (4.5) for every possible ancestor for each trajectory. This can be improved to  $\mathcal{O}(N)$  using rejection sampling or Markov moves; for details and recommendations we refer to Dau and Chopin (2023).

### 4.3 Advanced SMC methods

We conclude this chapter by discussing two advanced SMC methods that are required to understand the methodological contributions in Publications I and IV.

#### 4.3.1 The resample-move algorithm

In Section 4.2.2, we remarked that resampling is only a partial solution to the problem of degeneracy, because it does not create new particles. The *resample-move* algorithm (Gilks and Berzuini, 2001) addresses this limitation by augmenting resampling with a subsequent *move* step: apply, independently to each resampled path, a Markov kernel that leaves the current target invariant.

Formally, at time  $t+1$  of Algorithm 2, after resampling (line 6) we have the particle approximation

$$\tilde{\mathbb{Q}}_t^N(\mathbf{d}x_{0:t}) = \frac{1}{N} \sum_{n=1}^N \delta_{X_{0:t}^{A_{t+1}}}(x_{0:t}).$$

Now, let  $K_t(\mathbf{d}x'_{0:t} | x_{0:t})$  be a  $\mathbb{Q}_t$ -invariant kernel, meaning it satisfies

$$\int K_t(\mathbf{d}x'_{0:t} | x_{0:t}) \mathbb{Q}_t(\mathbf{d}x_{0:t}) = \mathbb{Q}_t(\mathbf{d}x'_{0:t}).$$

Invariance ensures that if we have a sample  $X_{0:t} \sim \mathbb{Q}_t$ , then a sample  $X'_{0:t} \sim K_t(\mathbf{d}x'_{0:t} | X_{0:t})$  obtained by sampling from  $K_t$  is also distributed as  $\mathbb{Q}_t$ . Applying  $K_t$  independently to each resampled path yields a new particle approximation

$$\hat{\mathbb{Q}}_t^N(\mathbf{d}x_{0:t}) := \frac{1}{N} \sum_{n=1}^N \delta_{\hat{X}_{0:t}^n}(\mathbf{d}x_{0:t}), \quad \hat{X}_{0:t}^n \sim K_t(\mathbf{d}x'_{0:t} | X_{0:t}^{A_{t+1}}),$$

which preserves the target and replaces duplicates, thereby increasing particle diversity.

A common choice for  $K_t$  is a Metropolis–Hastings kernel (Metropolis et al., 1953; Hastings, 1970). Given a current path  $X_{0:t}$ , we propose  $X'_{0:t} \sim \eta(\mathrm{d}x'_{0:t} \mid X_{0:t})$  from a user-specified proposal kernel  $\eta$  and accept with probability

$$\min \left( 1, \frac{\mathbb{Q}_t(X'_{0:t})\eta(X_{0:t} \mid X'_{0:t})}{\mathbb{Q}_t(X_{0:t})\eta(X'_{0:t} \mid X_{0:t})} \right),$$

otherwise we keep  $X_{0:t}$ . Since applying the move step to the full trajectory  $X_{0:t}$  has  $\mathcal{O}(t)$  cost, using resample-move at every time step results in overall  $\mathcal{O}(T^2)$  complexity for the particle filter, which can be prohibitively expensive. To mitigate this, we can rejuvenate only a fixed-lag window  $X_{t-\ell:t}$  while keeping  $X_{0:t-\ell}$  fixed, where  $\ell \ll t$  is a chosen lag. This reduces complexity to  $\mathcal{O}(T)$  and is particularly useful, for example, when only marginal filtering distributions are needed (Gilks and Berzuini, 2001). Another trick to reduce the average computational complexity is to only apply resample-move occasionally, either with a fixed probability or when some degeneracy criterion is met.

### 4.3.2 Iterated Batch Importance Sampling

We now consider an example of using SMC for parameter estimation by treating static parameters as latent variables. This is a good use case for the resample-move algorithm where it is worth paying the extra computational cost to alleviate degeneracy.

Consider a sequence of random variables  $(X_t)_{t=0}^T$  and a static parameter  $\Theta$  with joint distribution

$$\mathbb{P}(\mathrm{d}x_{0:t}, \mathrm{d}\theta) := \mu(\mathrm{d}\theta) \mathbb{P}_0(\mathrm{d}x_0) \prod_{s=1}^t P_s(\mathrm{d}x_s \mid x_{0:s-1}, \theta), \quad t \in [0 : T].$$

We wish to approximate the filtering distributions  $\mathbb{Q}_t(\mathrm{d}\theta_t) := \mathbb{P}(\mathrm{d}\theta \mid X_{0:t})$ . Note that this can be trivially written as a Feynman–Kac model for  $\Theta$ ,

$$\mathbb{Q}_t(\mathrm{d}\theta_{0:t}) = \mu(\mathrm{d}\theta_0) \mathbb{P}_0(x_0) \prod_{s=1}^t \underbrace{\delta_{\theta_{s-1}}(\mathrm{d}\theta_s)}_{:=M_s(\mathrm{d}\theta_s \mid \theta_{0:s-1})} \underbrace{P_s(x_s \mid x_{0:s-1}, \theta_s)}_{:=G_s(\theta_{0:s})}. \quad (4.6)$$

Observe that the transition kernels are Dirac measures (since the parameter is static) and the potential functions are the transition densities for  $(X_t)_{t=0}^T$ . A bootstrap particle filter for (4.6) would sample  $\theta_0^n \sim \mu$  for  $n \in [1 : N]$ , and then apply successive reweighting steps. However, if  $\mathbb{Q}_t$  is substantially different from  $\mu$ , then the support points  $\theta_0^{1:N}$  will be ill-suited to form a particle approximation of  $\mathbb{Q}_t$ , yielding poor approximations. The *iterated batch importance sampling (IBIS)* algorithm of Chopin (2002) proposes a solution by using resample-move steps to rejuvenate the  $\theta$ -particles whenever necessary. The full algorithm is given in Algorithm 3, which uses a  $\mathbb{Q}_t(\mathrm{d}\theta_t)$ -invariant Markov kernel  $K_t(\cdot \mid \theta_t)$  for the resample-move step.

---

**Algorithm 3:** Iterated batch importance sampling (IBIS, Chopin, 2002).

---

**input** : The Feynman–Kac model  $(Q_t)_{t=0}^T$  (4.6), kernels  $(K_t)_{t=0}^T$ , and number of particles  $N$ .

**output** : Particle approximation  $Q_t^N(d\theta_t) = \sum_{n=1}^N W_t^n \delta_{\Theta_t^n}(d\theta_t)$  and normalizing constant estimate  $L_t^N$  for  $t \in [0 : T]$ .

- 1  $\triangleright$  Operations involving the index  $n$  must be performed for  $n \in [1 : N]$ .
  - 2 Sample  $\Theta_0^n \sim \mu(d\theta_0)$
  - 3 Set  $w_0^n \leftarrow \mathbb{P}_0(x_0)$  and  $W_0^n \leftarrow w_0^n / \sum_{m=1}^N w_0^m$
  - 4 **for**  $t \in [1 : T]$  **do**
  - 5     **if** *Some degeneracy criterion is fulfilled* **then**
  - 6          $\triangleright$  Perform resample-move
  - 7         Sample  $A_t^n \sim \mathcal{C}(W_{t-1}^{1:N})$
  - 8         Sample  $\Theta_t^n \sim K_t(d\theta_t | \Theta_{t-1}^{A_t^n})$
  - 9         Set  $w_{t-1}^n \leftarrow 1$
  - 10     **else**
  - 11         Set  $\Theta_t^n \leftarrow \Theta_{t-1}^n$ .
  - 12     Set  $\alpha_t^n \leftarrow P_t(x_t | x_{0:t-1}, \Theta_t^n)$ ,  $w_t^n \leftarrow w_{t-1}^n \alpha_t^n$  and  $W_t^n \leftarrow w_t^n / \sum_{m=1}^N w_t^m$
-

# 5. Bayesian ODE Solvers and Parallel-in-Time State Estimation

This chapter treats two important topics that underpin Publication III:

1. Probabilistic ODE solving, in which the solution to an ODE is represented as a conditional distribution over functions and computed efficiently with Bayesian filters and smoothers.
2. Temporal parallelization of Bayesian filtering and smoothing, which yields algorithms that can readily make use of the massive opportunities for parallelism provided by modern graphics processing units (GPUs).

Together, they enable us to construct GPU-accelerated algorithms for probabilistic solutions to ODEs (and, by extension, partial differential equations), which constitute the main contribution of Publication III.

## 5.1 ODE solving as Bayesian inference

Consider the ordinary differential equation (ODE)

$$Dy(t) = f(t, y(t)), \quad y(0) = y_0, \tag{5.1}$$

defined on a time domain  $\mathbb{T} := [0, T]$  with  $T < \infty$ , where  $y_0 \in \mathbb{R}^d$  is the initial condition,  $f : \mathbb{T} \times \mathbb{R}^d \rightarrow \mathbb{R}^d$ , and  $D$  denotes the time derivative operator. Given a discretized time grid  $\mathbb{T}_N := \{t_0, t_1, \dots, t_N\}$  with  $0 = t_0 < \dots < t_N = T$ , classical numerical ODE solvers return point estimates  $\hat{y}_n := \hat{y}(t_n)$  of the solution for  $n \in [0 : N]$  (see, e.g., Butcher, 2016).

Here we consider a different problem, that of finding a *distribution* over possible solutions. The motivation for considering such probabilistic solutions is that they can accurately quantify the uncertainty induced by the time discretization performed to solve the ODE (Cockayne et al., 2019). This uncertainty can then be propagated for downstream tasks such as decision making, and can also be easily combined with uncertainty stemming from other sources, such as partial or noisy observations.

Adopting a Bayesian perspective, let  $Y := (Y_t)_{t \in \mathbb{T}}$  be a stochastic process defined on a probability space  $(Y, \mathcal{Y}, \mathbb{P})$ , where  $Y$  is a suitable class of functions in  $\mathbb{R}^d$  (e.g., absolutely continuous paths so that  $DY_t$  exists almost everywhere). A realization  $Y = y$  is to be interpreted as a candidate solution to (5.1), and our prior modeling choices are encoded in the law  $\mathbb{P}$  of  $Y$ . Information from the ODE is incorporated by conditioning on the following requirements for a valid solution:

$$\begin{aligned} Y_{t_0} &= y_0, \\ DY_{t_n} - f(t_n, Y_{t_n}) &= 0, \quad \forall n \in [0 : N], \end{aligned}$$

Let  $S$  denote the set of paths that meet these constraints. The updated distribution over trajectories is then the conditional probability distribution  $\mathbb{P}(\cdot | S)$ ,<sup>1</sup> which concentrates mass on paths consistent with the initial condition and the ODE evaluations on the grid.

The solution framework presented above for ODEs belongs to the class of *Bayesian probabilistic numerical* (BPN) methods, a formal definition of which can be found in Cockayne et al. (2019). In the following section, we describe a specific instantiation of this method where we choose a stochastic differential equation (SDE) prior for  $Y$ .

### 5.1.1 Stochastic differential equation priors

The ensuing presentation of an SDE prior for  $Y$  is adapted from Tronarp et al. (2021). Let

$$X_t^\top = \left( Y_t^\top \quad DY_t^\top \quad \dots \quad D^\nu Y_t^\top \right)$$

be a stochastic process constructed by stacking  $Y_t$  with its time derivatives up to order  $\nu \in \mathbb{N}$ , so  $X_t \in \mathbb{R}^{d(\nu+1)}$ . Now let  $\{e_m\}_{m=0}^\nu$  be the standard basis in  $\mathbb{R}^{\nu+1}$  and define the selection matrices  $E_m := e_m \otimes \mathbb{I}_d$ , where  $\otimes$  denotes the Kronecker product and  $\mathbb{I}_d$  is the identity matrix in  $\mathbb{R}^d$ . Notice that the matrix  $E_m$  picks out the  $m$ -th derivative of  $Y_t$  from  $X_t$ :

$$E_m^\top X_t = D^m Y_t, \quad m \in [0 : \nu], \quad (5.2)$$

where  $D^0 Y_t := Y_t$ . We then place the following linear time-invariant SDE prior for  $X_t$  (see, e.g., Øksendal, 2003):

$$dX_t = F X_t dt + E_\nu d\beta_t, \quad X_{t_0} \sim \mathcal{N}(0, \Sigma_0), \quad (5.3)$$

where  $\beta_t$  is a  $d$ -dimensional Brownian motion with diffusion matrix  $\Gamma \geq 0$  and  $F, \Sigma_0 \in \mathbb{R}^{d(\nu+1) \times d(\nu+1)}$  are some matrices ( $\Sigma_0 \geq 0$ ).

<sup>1</sup>Technical note: We assume that  $S$  is measurable and the conditional probability  $\mathbb{P}(\cdot | S)$  is well-defined (see, e.g., Williams, 1991, Sec. 9.9), which is satisfied for standard choices of the path space and priors.

To motivate this SDE prior, we note two complementary advantages—one conceptual and one computational. First, (5.3) places a Gaussian process (GP, Rasmussen and Williams, 2006) prior on  $Y_t$ , and the triplet  $(F, \Gamma, \Sigma_0)$  corresponds to a choice of the kernel function for the GP (Hartikainen and Särkkä, 2010; Lindgren et al., 2011). Classical GP kernels that admit such SDE representations include the  $\nu$ -times integrated Wiener process (IWP) and the Matérn family with half-integer smoothness  $(\nu+1/2)$ ; see Hartikainen and Särkkä (2010); Särkkä and Solin (2019) for detailed equivalences and more examples.

The second advantage is that the linear time-invariant SDE in (5.3) can be discretized to obtain linear-Gaussian transitions on any grid (Särkkä and Solin, 2019, Sec. 6.2):

$$(X_{t_{n+1}} | X_{t_n}) \sim \mathcal{N}(A(h_n)X_{t_n}, Q(h_n)),$$

where  $h_n := t_{n+1} - t_n$  and

$$A(h_n) := \exp(Fh_n),$$

$$Q(h_n) := \int_0^{h_n} A(h_n - \tau) E_\nu \Gamma E_\nu^\top A^\top(h_n - \tau) d\tau.$$

For standard choices such as IWP and half-integer Matérn priors,  $A(h_n)$  and  $Q(h_n)$  can be computed analytically.

### 5.1.2 The state-space model formulation

The stochastic process  $(X_{t_n})_{n=0}^N$  obtained by discretizing the SDE prior (5.3) is a linear-Gaussian Markov chain. The ODE constraint (5.1) on the grid  $\mathbb{T}_N$  can be written as

$$0 = DY_{t_n} - f(t_n, Y_{t_n}) = E_1^\top X_{t_n} - f(t_n, E_0^\top X_{t_n}) =: h(X_{t_n}),$$

where we used the property (5.2). We can interpret  $h(x)$  as an observation model and 0 as the value of an "observation"  $Z_{t_n}$  that we receive at "time"  $t_n$ . Taken together with the prior law of  $X_{t_n}$ , we have a state-space model (Section 2.1.1)

$$(X_{t_{n+1}} | X_{t_n}) \sim \mathcal{N}(A(h_n)X_{t_n}, Q(h_n)),$$

$$(Z_{t_n} | X_{t_n}) \sim \delta_{h(X_{t_n})},$$

where  $\delta_X$  is the Dirac measure centered at  $X$ . The Dirac measure can be viewed as a (degenerate) Gaussian with zero covariance (Bogachev, 1998), so this remains a (possibly singular) Gaussian state-space model. The posterior distribution  $p(X_{t_0:t_N} | z_{t_0:t_N} = 0)$  can now be obtained with, for example, the extended Kalman filter and smoother (since  $f$  may be nonlinear) or the iterated extended Kalman smoother (Section 2.3).

**Remark 5.1.** *In Publication III, we develop a Bayesian probabilistic numerical method for time-dependent partial differential equations (PDEs) that computes a maximum a posteriori (MAP) trajectory. The approach first converts a PDE into a set of coupled ODEs using the method of lines (Schiesser, 2012), and then uses the framework above to cast PDE solving as pathwise smoothing in a (Gaussian) state-space model, solved with the IEKS.*

## 5.2 Temporal parallelization of Bayesian smoothers

By construction, the Kalman filter and the RTS smoother described in Section 2.2 are *sequential* algorithms—the Kalman filter, for example, computes  $p(x_t | y_{0:t})$  from previously computed  $p(x_{t-1} | y_{0:t-1})$ . The consequence is that these algorithms have a *span complexity* of  $\mathcal{O}(T)$  for  $T$  time steps, where span complexity is defined as the maximum number of steps that need to be executed sequentially by a (possibly multithreaded) algorithm (Cormen et al., 2009, Ch. 27). As single-core clock speeds have largely plateaued (Suomela, 2025), modern accelerators deliver performance primarily through massive parallelism (e.g., graphics processing units (GPUs) expose tens of thousands of concurrent threads), leaving purely sequential algorithms unable to exploit available hardware.

To address this bottleneck, in this section we present a *parallel-in-time* formulation of the Kalman filter and smoother (Särkkä and García-Fernández, 2021), based on the parallel prefix-sums operation (Blelloch, 1990).

### 5.2.1 Prefix-sums

Let  $\{a_1, \dots, a_T\} \subseteq S$  be a set of elements and  $\oplus$  be an associative binary operator on the space  $S$ . The *all-prefix-sums* operation, which returns the ordered set

$$\{a_1, (a_1 \oplus a_2), \dots, (a_1 \oplus a_2 \oplus \dots \oplus a_T)\},$$

is a common operation in a variety of algorithms (Blelloch, 1990). For example, if  $\oplus$  is the addition operation, then all-prefix-sums returns the cumulative sums.

The prefix-sums can be obtained with a simple for-loop at a span complexity of  $\mathcal{O}(T)$ . However, if there are at least  $T$  parallel workers available which can perform computation and memory accesses independently of each other, various algorithms exist which can perform all-prefix-sums at  $\mathcal{O}(\log T)$  span complexity (Pibiri and Venturini, 2021), with an example given in Algorithm 4. For simplicity, the algorithm assumes that  $T$  is a power of two; otherwise one pads the input with identity elements up to the next power of two. High-quality implementations are available in modern programming libraries like CUDA C++ (`thrust::inclusive_scan`, NVIDIA, 2023) and JAX (`jax.numpy.associative_scan`, Bradbury et al., 2018).

---

**Algorithm 4:** Parallel scan for all-prefix-sums (Blelloch, 1990).

---

**input** : An ordered set  $\{a_1, \dots, a_T\}$  and associative binary operator  $\oplus$ .  
**output** : The ordered set  $\{a_1, (a_1 \oplus a_2), \dots, (a_1 \oplus a_2 \oplus \dots \oplus a_T)\}$ .

```

1 for  $i \leftarrow 1$  to  $T$  do                                // Compute in parallel
2    $b_i \leftarrow a_i$                                        // Save the input
3 for  $d \leftarrow 0$  to  $\log_2 T - 1$  do                        // Up sweep
4   for  $i \leftarrow 0$  to  $T - 1$  by  $2^{d+1}$  do                // Compute in parallel
5      $j \leftarrow i + 2^d$ 
6      $k \leftarrow i + 2^{d+1}$ 
7      $a_k \leftarrow a_j \oplus a_k$ 
8  $a_T \leftarrow I$  //  $I$  is the identity element with respect to  $\oplus$ .
9 for  $d \leftarrow \log_2 T - 1$  to  $0$  do                        // Down sweep
10  for  $i \leftarrow 0$  to  $T - 1$  by  $2^{d+1}$  do                // Compute in parallel
11   $j \leftarrow i + 2^d$ 
12   $k \leftarrow i + 2^{d+1}$ 
13   $t \leftarrow a_j$ 
14   $a_j \leftarrow a_k$ 
15   $a_k \leftarrow a_k \oplus t$ 
16 for  $i \leftarrow 1$  to  $T$  do                                // Final pass, compute in parallel
17    $a_i \leftarrow a_i \oplus b_i$ 

```

---

### 5.2.2 Bayesian filtering and smoothing as prefix-sums

We now address the problem of parallelizing the Bayesian filtering and smoothing equations for state-space models (Theorems 2.3 and 2.5) along the time dimension. Särkkä and García-Fernández (2021) recast the forward and backward recursions of Bayesian filtering and smoothing as parallel scans: per-time elements  $a_t$  and a binary operator  $\oplus$  are chosen so that the all-prefix-sums of  $(a_t)_{t=0}^T$  yields all filtering distributions  $p(x_t | y_{0:t})$  and (for another choice of  $a_t$  and  $\oplus$ ) the marginal smoothing distributions  $p(x_t | y_{0:T})$ . We state the concrete choices of  $(a_t, \oplus)$  for filtering and smoothing in the propositions below; proofs and derivations are given in Särkkä and García-Fernández (2021).

**Proposition 5.2** (Bayesian filtering as all-prefix-sums). *For  $t \in [0 : T]$ , define  $a_t := (f_t, g_t)$  where*

$$f_t(x_t | x_{t-1}) := p(x_t | y_t, x_{t-1}), \quad g_t(x_{t-1}) := p(y_t | x_{t-1}), \quad (5.4)$$

*with the convention  $f_0(x_0 | x_{-1}) := p(x_0 | y_0)$  and  $g_0(x_{-1}) := p(y_0)$ . Also define the operator  $(f_i, g_i) \oplus (f_j, g_j) := (f_{ij}, g_{ij})$  where*

$$f_{ij}(x | z) = \frac{\int g_j(y) f_j(x | y) f_i(y | z) dy}{\int g_j(y) f_i(y | z) dy}, \quad g_{ij}(z) = g_i(z) \int g_j(y) f_i(y | z) dy. \quad (5.5)$$

*Then, the operator  $\oplus$  is associative, and the  $t$ -th prefix-sum equals the filtering distribution at time  $t$  and the marginal likelihood,*

$$a_0 \oplus a_1 \oplus \cdots \oplus a_t = \begin{pmatrix} p(x_t | y_{0:t}) \\ p(y_{0:t}) \end{pmatrix}.$$

**Proposition 5.3** (Bayesian smoothing as all-prefix-sums). *For  $t \in [0 : T]$ , define*

$$a_t := p(x_t | y_{0:t}, x_{t+1}), \quad (5.6)$$

*with the convention  $a_T := p(x_T | y_{0:T})$ . Then the binary operator  $\oplus$  defined as*

$$(a_i \oplus a_j)(x | z) := \int a_i(x | y) a_j(y | z) dy \quad (5.7)$$

*is associative and the  $t$ -th reverse prefix-sum,  $a_t \oplus a_{t+1} \oplus \cdots \oplus a_T$ , is the marginal smoothing distribution  $p(x_t | y_{0:T})$ .*

In general, neither the elements (5.4) and (5.6) nor the operations (5.5) and (5.7) can be computed in closed form, so Propositions 5.2 and 5.3 do not provide universal utility for state-space models. Nevertheless, there are two important exceptions which admit closed form expressions: (i) linear-Gaussian SSMs and (ii) SSMs with finite state and observation spaces. In these cases, the parallel scan (Algorithm 4) yields exact filtering and

smoothing distributions with  $\mathcal{O}(\log T)$  span and  $\mathcal{O}(T)$  work (Särkkä and García-Fernández, 2021; Hassan et al., 2021).

The parallel-in-time approach extends naturally to the iterated extended Kalman smoother (Yaghoobi et al., 2021). Within each IEKS iteration, linearizations at different time points depend only on the current trajectory and are therefore embarrassingly parallel; the resulting linear-Gaussian subproblem is then solved with the same scan-based smoother. In practice, numerical stability is improved by square-root implementations that propagate Cholesky factors of covariance matrices. Such square-root filters and smoothers have also been parallelized and are typically the preferred choice for practitioners (Yaghoobi et al., 2025).

**Remark 5.4.** *In Publication III, we used the parallel implementation of the IEKS to obtain the MAP estimate of nonlinear time-dependent PDEs in  $\mathcal{O}(\log T)$  span complexity.*



## 6. Summary and Discussion

We conclude this thesis by summarizing the contributions of the publications that comprise it. For each publication, we present a concise overview of the method, discuss the main findings, and reflect on how the work relates to the existing literature and directions for future research.

### 6.1 Publication I: Nesting particle filters for experimental design in dynamical systems

In this work, we consider the problem of learning an optimal policy for sequential Bayesian experimental design (Section 3.4.2). Formally, let  $\pi$  be a stochastic policy, and let the joint distribution of designs  $\xi_{1:T}$  and observations  $y_{1:T}$  conditional on hidden parameters  $\theta$  be given by

$$p_\pi(\xi_{1:T}, y_{1:T} | \theta) = \prod_{t=1}^T \pi(\xi_t | \xi_{1:t-1}, y_{1:t-1}) f_\theta(y_t | y_{t-1}, \xi_t), \quad \xi_{1:0} = y_{1:0} := \emptyset.$$

The goal is to learn a policy that maximizes the expected information gain (EIG) over the full sequence of experiments (3.17):

$$U(\pi) := \mathbb{H}[\Theta] - \mathbb{H}[\Theta | Y_{1:T}, \Xi_{1:T}].$$

Learning a policy before performing experiments enables real-time experimentation, since minimal computational effort is required to select each subsequent design.

**The method.** We first observed that the EIG can be written as an expected cumulative reward over the experiments:

$$U(\pi) = \mathbb{E}_{p_\pi} \left[ \sum_{t=1}^T R_t(\Xi_{1:t}, Y_{1:t}) \right],$$

where  $R_t$  is defined in Proposition 3.5, and requires computing expectations with respect to  $p(\theta | \xi_{1:t}, y_{1:t})$ . Then, using the general learning formulation

from Section 3.2, we framed policy optimization as a maximum likelihood estimation problem in a Feynman–Kac model of the form

$$q_{\pi}(\xi_{1:T}, y_{1:T}) \propto p_{\pi}(\xi_{1:T}, y_{1:T}) \prod_{t=1}^T \exp\{\mathcal{R}_t(\xi_{1:t}, y_{1:t})\}.$$

The main computational challenge lies in sampling from this FK model, for which we developed a nested sequential Monte Carlo algorithm called Inside–Out SMC<sup>2</sup>. This takes inspiration from the SMC<sup>2</sup> algorithm of Chopin et al. (2013), which nests a standard particle filter inside IBIS (Section 4.3.2) for parameter estimation in state-space models.

**Main findings.** The method we introduced is a generic policy optimization algorithm for sequential Bayesian experimental design which can handle nonlinear, non-Gaussian observation likelihoods  $f_{\theta}$ . Empirically, our learned policies match or outperform state-of-the-art algorithms that also assume  $f_{\theta}$  is known (Foster et al., 2021). In addition, Inside–Out SMC<sup>2</sup> can be used to estimate the EIG using far fewer samples compared to commonly used EIG surrogates (Foster et al., 2019).

**Reflections and outlook.** The algorithm we introduced provides a compelling alternative to existing methods for training policies in sequential BED. However, a key limitation is the requirement that the transition density  $f_{\theta}$  be available in closed form. This assumption does not hold in some practical scenarios where simulation from the observation model is feasible but density evaluation is not (Ivanova et al., 2021), thereby limiting the method’s applicability. A natural extension is therefore to remove this limitation by developing an algorithm that learns completely from samples alone (Blau et al., 2022). Another limitation, which we addressed in Publication IV, is that Inside–Out SMC<sup>2</sup> has  $\mathcal{O}(T^2)$  time complexity, which may make it prohibitively expensive for designing very long sequences of experiments.

## 6.2 Publication II: Sequential Monte Carlo for policy optimization in continuous POMDPs

This work addresses the problem of optimal control of partially observable Markov decision processes with continuous state, observation, and action spaces (continuous POMDPs). As highlighted in Section 3.3, learning optimal policies for continuous POMDPs is challenging because the value function depends on the infinite-dimensional belief state (i.e., the filtering distribution of the hidden state).

**The method.** The proposed method is a policy gradient algorithm (Sutton and Barto, 2018, Ch. 13) for finite-horizon continuous POMDPs based on

the general inference and learning framework presented in Section 3.2. The gradient of the objective is computed as an integral under a Feynman-Kac model which assigns higher probability density to observation-action trajectories that produce high returns (3.14). We introduced a nested SMC algorithm that enables us to sample from the FK model and estimate the gradient.

**Main findings.** Our algorithm either matches or outperforms state-of-the-art deep reinforcement learning algorithms on selected tasks with continuous POMDPs. In particular, in POMDPs in which actions exhibit the dual effect (actions influence returns *and* state estimates), our algorithm demonstrates a clear advantage over competitors by explicitly incorporating the value of information gathering. Our method also generalizes classical belief-space POMDP planners without relying on linear or Gaussian approximations (Van Den Berg et al., 2012), similar to Thrun (1999).

**Reflections and outlook.** We introduced a principled policy optimization algorithm that operates directly on the belief space. We do not make any simplifying assumptions about the structure of the decision-making process, and the algorithm’s effectiveness is primarily limited by available computational resources. This contrasts with much prior work on POMDPs, which typically relies on assumptions that limit the agent’s ability to explicitly account for the information-gathering outcome of actions (Littman et al., 1995).

We sample from the Feynman-Kac model by effectively nesting one bootstrap particle filter inside another. However, when scaling to high dimensions, it would likely be necessary to either learn better proposals for SMC (see, e.g., Naesseth et al., 2018) or use *twisted SMC* (Guarniero et al., 2017) to ensure sample efficiency. Another limitation is that our method requires the observation density to be tractable, since we use a particle filter to keep track of the hidden state. Replacing this particle filter with a state estimator that learns from samples would make the algorithm more widely applicable.

### 6.3 Publication III: Parallel-in-time probabilistic solutions for time-dependent nonlinear partial differential equations

This work introduces a Bayesian probabilistic numerical method (Cockayne et al., 2019) for solving time-dependent partial differential equations. We consider PDEs of the form

$$\frac{\partial u}{\partial t}(t, x) = f(t, x, u(t, x), \mathcal{D}u(t, x)), \quad t \in [0, T], \quad x \in [a, b],$$

subject to initial and boundary conditions

$$u(0, x) = h(x), \quad \forall x \in [a, b],$$

$$u(t, x) = g(x), \quad \forall (t, x) \in [0, T] \times \{a, b\}.$$

Here  $f$ ,  $g$  and  $h$  are known nonlinear functions and  $\mathcal{D}$  is a differential operator.

**The method.** We first convert the PDE into a set of coupled ODEs using the method of lines (MOL, Schiesser, 2012), which discretizes all spatial dimensions while leaving time continuous. Solving these coupled ODEs is then formulated as a MAP estimation problem in a nonlinear state-space model (Section 5.1), and we compute the MAP estimate using the iterated extended Kalman smoother (Section 2.3.1). The advantage of the SSM formulation is that it allows us to leverage parallel-in-time methods to compute the solution (Section 5.2), enabling efficient computation on modern parallel architectures.

**Main findings.** When compared to BPN methods that rely on the extended Kalman filter to estimate the solution (Krämer et al., 2022), our approach achieves comparable accuracy in the posterior mean estimates, as measured by the  $L_2$  error with respect to a reference solution, while also providing better calibrated posterior uncertainty, as measured by the log density of the reference trajectory under the posterior. Another major benefit is that we achieve  $\mathcal{O}(\log N)$  span complexity for the algorithm, where  $N$  is the number of discretization points in time. On hardware accelerators such as GPUs, this allows us to use finer time grids—and achieve higher-fidelity solutions—in comparable wall-clock time to serial methods with coarser grids.

**Reflections and outlook.** A limitation of our approach is that it does not account for discretization error from the method of lines. The uncertainty quantification therefore captures only temporal discretization error, not spatial discretization error. This could be addressed by incorporating probabilistic spatial discretization methods such as those developed by Krämer et al. (2022), which would provide a fully probabilistic treatment of both spatial and temporal approximations.

## 6.4 Publication IV: Recursive nested filtering for efficient amortized Bayesian experimental design

In Publication I, we introduced the Inside-Out SMC<sup>2</sup> algorithm to learn policies for sequential BED. This nested SMC algorithm makes use of resample-move steps (Section 4.3.1) to rejuvenate the  $\theta$ -particles tracking the parameter posteriors for each simulated experiment. Because each move step after  $t$  experiments has  $\mathcal{O}(t)$  complexity, Inside-Out SMC<sup>2</sup> has overall complexity  $\mathcal{O}(NMT^2)$ , where  $N$  is the number of simulated experimental histories,  $M$  is the number of  $\theta$ -particles per posterior, and  $T$  is the number of experiments.

This quadratic scaling in  $T$  limits applicability to long sequences, which we address in Publication IV.

**The method.** In this work, we used an alternative to the traditional move step known as *jittering* (Liu and West, 2001). Instead of using a target-invariant Markov kernel to rejuvenate the  $\theta$ -particles, we perturb the particles using a kernel with  $\mathcal{O}(1)$  cost. While the perturbed particles are no longer distributed according to the filtering distribution, by carefully controlling the magnitude of the induced perturbation as a function of the number of samples  $M$ , we can still guarantee consistency of estimates while eliminating the prohibitive linear cost of the move step. This approach was inspired by the *nested particle filter* (NPF) of Crisan and Míguez (2017, 2018), which analogously aims to improve the computational complexity of SMC<sup>2</sup>. The resulting algorithm, called *Inside-Out NPF*, also incorporates an optional backward sampling procedure to improve the diversity of samples.

**Main findings.** Inside-Out NPF achieves  $\mathcal{O}(NMT)$  computational complexity, enabling our learning framework developed in Publication I to scale to settings with long experiment sequences. We conducted a theoretical analysis of the particle approximations produced by the Inside-Out NPF, and proved consistency of integral estimates under reasonable assumptions on the jittering kernel. The algorithm also performs competitively with Inside-Out SMC<sup>2</sup> while being significantly faster.

**Reflections and outlook.** The Inside-Out NPF is a fully recursive algorithm to find optimal trajectories for sequential BED. However, it shares the same weakness as Inside-Out SMC<sup>2</sup>: the observation density must be tractable. Additionally, using backward sampling with Inside-Out NPF increases the computational complexity back to being quadratic in time, negating the benefit of the jittering kernel. Future improvements could explore alternative jittering kernel designs and investigate additional theoretical properties beyond consistency, such as non-asymptotic bounds.



# References

- Abdulsamad, H., Iqbal, S., Corenflos, A., and Särkkä, S. (2023). Risk-sensitive stochastic optimal control as Rao-Blackwellized Markovian score climbing. *arXiv preprint arXiv:2312.14000*. Cited on page 29.
- Abdulsamad, H., Iqbal, S., and Särkkä, S. (2025). Sequential Monte Carlo for policy optimization in continuous POMDPs. In *Advances in Neural Information Processing Systems*. Cited on page 38.
- Agapiou, S., Papaspiliopoulos, O., Sanz-Alonso, D., and Stuart, A. M. (2017). Importance sampling: Intrinsic dimension and computational cost. *Statistical Science*, 32(3):405–431. Cited on page 48.
- Amos, B. (2023). Tutorial on amortized optimization. *Foundations and Trends® in Machine Learning*, 16(5):592–732. Cited on page 30.
- Aoki, M. (1967). *Optimization of Stochastic Systems: Topics in Discrete-Time Systems*. Mathematics in Science and Engineering. Academic Press. Cited on page 36.
- Åström, K. J. (1965). Optimal control of Markov processes with incomplete state information. *Journal of Mathematical Analysis and Applications*, 10(1):174–205. Cited on page 36.
- Attias, H. (2003). Planning by probabilistic inference. In *Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics*. Cited on page 31.
- Bar-Shalom, Y. and Tse, E. (1974). Dual effect, certainty equivalence, and separation in stochastic control. *IEEE Transactions on Automatic Control*, 19(5):494–500. Cited on page 38.
- Bell, B. M. (1994). The iterated Kalman smoother as a Gauss-Newton method. *SIAM Journal on Optimization*, 4(3):626–636. Cited on page 26.
- Bellman, R. (1957a). A Markovian decision process. *Journal of Mathematics and Mechanics*, 6(5):679–684. Cited on page 30.
- Bellman, R. E. (1957b). *Dynamic Programming*. Princeton University Press. Cited on page 30.
- Berger, J. O. (1985). *Statistical Decision Theory and Bayesian Analysis*. Springer New York, 2nd edition. Cited on page 19.
- Bernardo, J. M. (1979). Expected information as expected utility. *The Annals of Statistics*, 7(3):686–690. Cited on page 41.

- Bernardo, J. M. and Smith, A. F. M. (1994). *Bayesian Theory*. John Wiley & Sons. Cited on pages 15, 16, 19, and 39.
- Bertsekas, D. (2012). *Dynamic Programming and Optimal Control: Volume I*. Athena Scientific, 4th edition. Cited on page 30.
- Blau, T., Bonilla, E. V., Chades, I., and Dezfouli, A. (2022). Optimizing sequential experimental design with deep reinforcement learning. In *International Conference on Machine Learning*. Cited on pages 42 and 64.
- Blei, D. M., Kucukelbir, A., and McAuliffe, J. D. (2017). Variational inference: A review for statisticians. *Journal of the American Statistical Association*, 112(518):859–877. Cited on page 33.
- Blelloch, G. E. (1990). Prefix sums and their applications. Technical Report CMU-CS-90-190, School of Computer Science, Carnegie Mellon University. Cited on pages 58 and 59.
- Bogachev, V. I. (1998). *Gaussian Measures*. American Mathematical Society. Cited on page 57.
- Bradbury, J., Frostig, R., Hawkins, P., Johnson, M. J., Leary, C., Maclaurin, D., Necula, G., Paszke, A., VanderPlas, J., Wanderman-Milne, S., and Zhang, Q. (2018). JAX: composable transformations of Python+NumPy programs. Cited on pages 26 and 58.
- Briers, M., Doucet, A., and Maskell, S. (2010). Smoothing algorithms for state-space models. *Annals of the Institute of Statistical Mathematics*, 62(1):61–89. Cited on page 51.
- Butcher, J. C. (2016). *Numerical Methods for Ordinary Differential Equations*. John Wiley & Sons. Cited on page 55.
- Cappé, O., Moulines, E., and Rydén, T. (2005). *Inference in Hidden Markov Models*. Springer. Cited on pages 20, 34, and 35.
- Carpenter, J., Clifford, P., and Fearnhead, P. (1999). An improved particle filter for nonlinear problems. *IEE Proceedings - Radar, Sonar and Navigation*, 146(1):2–7. Cited on page 50.
- Cassandra, A., Littman, M. L., and Zhang, N. L. (1997). Incremental pruning: A simple, fast, exact method for partially observable Markov decision processes. In *Conference on Uncertainty in Artificial Intelligence*. Cited on page 37.
- Chaloner, K. and Verdinelli, I. (1995). Bayesian experimental design: A review. *Statistical Science*, 10(3):273–304. Cited on page 40.
- Chatterjee, S. and Diaconis, P. (2018). The sample size required in importance sampling. *The Annals of Applied Probability*, 28(2):1099–1135. Cited on page 48.
- Chopin, N. (2002). A sequential particle filter method for static models. *Biometrika*, 89(3):539–552. Cited on pages 43, 53, and 54.
- Chopin, N., Jacob, P. E., and Papaspiliopoulos, O. (2013). SMC<sup>2</sup>: An efficient algorithm for sequential analysis of state space models. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 75(3):397–426. Cited on page 64.
- Chopin, N. and Papaspiliopoulos, O. (2020). *An Introduction to Sequential Monte Carlo*. Springer. Cited on pages 27, 46, 49, and 50.
- Chopin, N., Singh, S. S., Soto, T., and Vihola, M. (2022). On resampling schemes for particle filters with weakly informative observations. *The Annals of Statistics*, 50(6):3197–3222. Cited on page 50.

- Cockayne, J., Oates, C. J., Sullivan, T. J., and Girolami, M. (2019). Bayesian probabilistic numerical methods. *SIAM Review*, 61(4):756-789. Cited on pages 55, 56, and 65.
- Cormen, T. H., Leiserson, C. E., Rivest, R. L., and Stein, C. (2009). *Introduction to Algorithms*. MIT Press, 3rd edition. Cited on page 58.
- Cox, H. (1964). On the estimation of state variables and parameters for noisy dynamic systems. *IEEE Transactions on Automatic Control*, 9(1):5-12. Cited on page 26.
- Crisan, D. (2001). Particle filters - A theoretical perspective. In Doucet, A., Freitas, N., and Gordon, N., editors, *Sequential Monte Carlo Methods in Practice*. Springer. Cited on page 50.
- Crisan, D. and Doucet, A. (2002). A survey of convergence results on particle filtering methods for practitioners. *IEEE Transactions on Signal Processing*, 50(3):736-746. Cited on page 50.
- Crisan, D. and Míguez, J. (2017). Uniform convergence over time of a nested particle filtering scheme for recursive parameter estimation in state-space Markov models. *Advances in Applied Probability*, 49(4):1170-1200. Cited on page 67.
- Crisan, D. and Míguez, J. (2018). Nested particle filters for online parameter estimation in discrete-time state-space Markov models. *Bernoulli*, 24(4a):3039-3086. Cited on page 67.
- Dau, H.-D. and Chopin, N. (2023). On backward smoothing algorithms. *The Annals of Statistics*, 51(5):2145 - 2169. Cited on page 52.
- Dayan, P. and Hinton, G. E. (1997). Using expectation-maximization for reinforcement learning. *Neural Computation*, 9(2):271-278. Cited on pages 29 and 31.
- Del Moral, P. (2004). *Feynman-Kac Formulae: Genealogical and Interacting Particle Systems with Applications*. Springer. Cited on pages 27 and 50.
- Del Moral, P., Doucet, A., and Jasra, A. (2012). On adaptive resampling strategies for sequential Monte Carlo methods. *Bernoulli*, 18(1):252-278. Cited on page 51.
- Douc, R., Cappé, O., and Moulines, E. (2005). Comparison of resampling schemes for particle filtering. In *Proceedings of the 4th International Symposium on Image and Signal Processing and Analysis*, pages 64-69. Cited on page 50.
- Douc, R., Garivier, A., Moulines, E., and Olsson, J. (2011). Sequential Monte Carlo smoothing for general state space hidden Markov models. *The Annals of Applied Probability*, 21(6):2109-2145. Cited on page 51.
- Doucet, A., de Freitas, N., and Gordon, N., editors (2001). *Sequential Monte Carlo Methods in Practice*. Springer. Cited on pages 16 and 50.
- Doucet, A. and Johansen, A. M. (2011). A tutorial on particle filtering and smoothing: Fifteen years later. In Crisan, D. and Rozovskii, B., editors, *The Oxford Handbook of Nonlinear Filtering*, chapter 24, pages 656-704. Oxford University Press. Cited on page 49.
- Elliott, C. (2018). The simple essence of automatic differentiation. *Proceedings of the ACM on Programming Languages*, 2(ICFP). Cited on page 26.
- Fisher, R. A. (1935). *The Design of Experiments*. Oliver and Boyd. Cited on page 39.
- Foster, A., Ivanova, D. R., Malik, I., and Rainforth, T. (2021). Deep adaptive design: Amortizing sequential Bayesian experimental design. In *International Conference on Machine Learning*. Cited on pages 42 and 64.

- Foster, A., Jankowiak, M., Bingham, E., Horsfall, P., Teh, Y. W., Rainforth, T., and Goodman, N. (2019). Variational Bayesian optimal experimental design. *Advances in neural information processing systems*, 32. Cited on pages 42 and 64.
- Gilks, W. R. and Berzuini, C. (2001). Following a moving target—Monte Carlo inference for dynamic Bayesian models. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 63(1):127–146. Cited on pages 52 and 53.
- Godsill, S. J., Doucet, A., and West, M. (2004). Monte Carlo smoothing for nonlinear time series. *Journal of the American Statistical Association*, 99(465):156–168. Cited on page 51.
- Gordon, N. J., Salmond, D. J., and Smith, A. F. M. (1993). Novel approach to nonlinear/non-Gaussian Bayesian state estimation. In *IEE proceedings F (Radar and Signal Processing)*, volume 140, pages 107–113. Cited on page 50.
- Griewank, A. and Walther, A. (2008). *Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation*. Society for Industrial and Applied Mathematics, second edition. Cited on page 26.
- Guarniero, P., Johansen, A. M., and Lee, A. (2017). The iterated auxiliary particle filter. *Journal of the American Statistical Association*, 112(520):1636–1647. Cited on page 65.
- Haarhoja, T., Zhou, A., Abbeel, P., and Levine, S. (2018). Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International Conference on Machine Learning*. Cited on page 33.
- Hafner, D., Lillicrap, T., Ba, J., and Norouzi, M. (2019). Dream to control: Learning behaviors by latent imagination. *arXiv preprint arXiv:1912.01603*. Cited on page 37.
- Hartikainen, J. and Särkkä, S. (2010). Kalman filtering and smoothing solutions to temporal Gaussian process regression models. In *IEEE International Workshop on Machine Learning for Signal Processing*, pages 379–384. Cited on page 57.
- Hassan, S. S., Särkkä, S., and García-Fernández, Á. F. (2021). Temporal parallelization of inference in hidden Markov models. *IEEE Transactions on Signal Processing*, 69:4875–4887. Cited on page 61.
- Hastings, W. K. (1970). Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1):97–109. Cited on page 53.
- Hoffman, M., de Freitas, N., Doucet, A., and Peters, J. (2009). An expectation maximization algorithm for continuous Markov decision processes with arbitrary rewards. In *Proceedings of the 12<sup>th</sup> International Conference on Artificial Intelligence and Statistics*. Cited on page 31.
- Huan, X., Jagalur, J., and Marzouk, Y. (2024). Optimal experimental design: Formulations and computations. *Acta Numerica*, 33:715–840. Cited on page 39.
- Huan, X. and Marzouk, Y. M. (2016). Sequential bayesian optimal experimental design via approximate dynamic programming. *arXiv preprint arXiv:1604.08320*. Cited on page 42.
- Iqbal, S., Corenflos, A., Särkkä, S., and Abdulsamad, H. (2024). Nesting particle filters for experimental design in dynamical systems. In *International Conference on Machine Learning*. Cited on page 42.
- Ivanova, D. R., Foster, A., Kleinegesse, S., Gutmann, M., and Rainforth, T. (2021). Implicit deep adaptive design: Policy-based experimental design without likelihoods. In *Advances in Neural Information Processing Systems*. Cited on pages 42 and 64.

- Jazwinski, A. H. (1970). *Stochastic Processes and Filtering Theory*. Academic Press. Cited on page 26.
- Kaelbling, L. P., Littman, M. L., and Cassandra, A. R. (1998). Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101(1):99-134. Cited on page 37.
- Kallenberg, O. (2021). *Foundations of Modern Probability*. Springer Cham, 3rd edition. Cited on page 47.
- Kappen, H. J. (2005). Path integrals and symmetry breaking for optimal control theory. *Journal of Statistical Mechanics: Theory and Experiment*, 2005(11):P11011. Cited on page 31.
- Kappen, H. J., Gómez, V., and Opper, M. (2012). Optimal control as a graphical model inference problem. *Machine learning*, 87(2):159-182. Cited on page 31.
- Kitagawa, G. (1996). Monte Carlo filter and smoother for non-Gaussian nonlinear state space models. *Journal of Computational and Graphical Statistics*, 5(1):1-25. Cited on page 51.
- Krämer, N., Schmidt, J., and Hennig, P. (2022). Probabilistic numerical method of lines for time-dependent partial differential equations. In *International Conference on Artificial Intelligence and Statistics*, volume 151, pages 625-639. Cited on page 66.
- Kueck, H., Hoffman, M., Doucet, A., and de Freitas, N. (2009). Inference and learning for active sensing, experimental design and control. In *Pattern Recognition and Image Analysis*. Springer Berlin Heidelberg. Cited on pages 15 and 29.
- Kullback, S. and Leibler, R. A. (1951). On information and sufficiency. *The Annals of Mathematical Statistics*, 22(1):79-86. Cited on page 33.
- Kálmán, R. E. (1960). A new approach to linear filtering and prediction problems. *Transactions of the ASME, Journal of Basic Engineering*, 82:35-45. Cited on pages 15 and 24.
- Lee, A. X., Nagabandi, A., Abbeel, P., and Levine, S. (2020). Stochastic latent actor-critic: Deep reinforcement learning with a latent variable model. In *Advances in Neural Information Processing Systems*. Cited on page 37.
- Levine, S. (2018). Reinforcement learning and control as probabilistic inference: Tutorial and review. Cited on pages 29, 31, and 33.
- Li, T., Beirami, A., Sanjabi, M., and Smith, V. (2023). On tilted losses in machine learning: Theory and applications. *Journal of Machine Learning Research*, 24(142):1-79. Cited on pages 32 and 35.
- Lindgren, F., Rue, H., and Lindström, J. (2011). An explicit link between Gaussian fields and Gaussian Markov random fields: The stochastic partial differential equation approach. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 73(4):423-498. Cited on page 57.
- Lindley, D. V. (1956). On a measure of the information provided by an experiment. *The Annals of Mathematical Statistics*, 27(4):986-1005. Cited on pages 39 and 40.
- Lindsten, F. and Schön, T. B. (2013). Backward simulation methods for Monte Carlo statistical inference. *Foundations and Trends® in Machine Learning*, 6(1):1-143. Cited on page 51.
- Littman, M. L., Cassandra, A. R., and Kaelbling, L. P. (1995). Learning policies for partially observable environments: Scaling up. In *International Conference on Machine Learning*. Cited on pages 37 and 65.

- Liu, J. and West, M. (2001). Combined parameter and state estimation in simulation-based filtering. In Doucet, A., Freitas, N., and Gordon, N., editors, *Sequential Monte Carlo Methods in Practice*. Springer. Cited on page 67.
- Liu, J. S. (1996). Metropolisized independent sampling with comparisons to rejection sampling and importance sampling. *Statistics and Computing*, 6:113-119. Cited on page 48.
- Liu, J. S. and Chen, R. (1995). Blind deconvolution via sequential imputations. *Journal of the American Statistical Association*, 90(430):567-576. Cited on pages 48 and 51.
- MacKay, D. J. (2003). *Information Theory, Inference and Learning Algorithms*. Cambridge University Press. Cited on page 40.
- Marcus, S. I., Fernández-Gaucherand, E., Hernández-Hernandez, D., Coraluppi, S., and Fard, P. (1997). Risk sensitive Markov decision processes. In *Systems and control in the twenty-first century*, pages 263-279. Birkhäuser Boston. Cited on page 32.
- Martin, G. M., Frazier, D. T., and Robert, C. P. (2023). Computing Bayes: From then 'til now. *Statistical Science*, pages 1-17. Cited on page 19.
- Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., and Teller, E. (1953). Equation of state calculations by fast computing machines. *The Journal of Chemical Physics*, 21(6):1087-1092. Cited on page 53.
- Metropolis, N. and Ulam, S. (1949). The Monte Carlo method. *Journal of the American Statistical Association*, 44(247):335-341. Cited on page 47.
- Naesseth, C., Linderman, S., Ranganath, R., and Blei, D. (2018). Variational sequential Monte Carlo. In *International Conference on Artificial Intelligence and Statistics*. Cited on page 65.
- Nocedal, J. and Wright, S. J. (2006). *Numerical Optimization*. Springer, 2nd edition. Cited on page 26.
- NVIDIA (2023). CUDA Toolkit Documentation 12.3. Cited on page 58.
- Øksendal, B. (2003). *Stochastic Differential Equations: An Introduction with Applications*. Springer, 6th edition. Cited on page 56.
- Owen, A. B. (2013). *Monte Carlo Theory, Methods and Examples*. <https://artowen.su.domains/mc/>. Cited on page 48.
- Pibiri, G. E. and Venturini, R. (2021). Practical trade-offs for the prefix-sum problem. *Software: Practice and Experience*, 51(5):921-949. Cited on page 58.
- Pineau, J., Gordon, G., and Thrun, S. (2003). Point-based value iteration: An anytime algorithm for POMDPs. In *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence*. Cited on page 37.
- Puterman, M. L. (2014). *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons. Cited on page 30.
- Rainforth, T., Foster, A., Ivanova, D. R., and Bickford Smith, F. (2024). Modern Bayesian experimental design. *Statistical Science*, 39(1):100-114. Cited on pages 41 and 42.
- Rasmussen, C. E. and Williams, C. K. I. (2006). *Gaussian Processes for Machine Learning*. The MIT Press. Cited on page 57.
- Rauch, H. E., Tung, F., and Striebel, C. T. (1965). Maximum likelihood estimates of linear dynamic systems. *AIAA Journal*, 3(8):1445-1450. Cited on page 25.

- Rawlik, K., Toussaint, M., and Vijayakumar, S. (2012). On stochastic optimal control and reinforcement learning by approximate inference. In *Proceedings of Robotics: Science and Systems VIII*. Cited on pages 29, 31, and 33.
- Rawlik, K. C. (2013). *On probabilistic inference approaches to stochastic optimal control*. PhD thesis, University of Edinburgh. Cited on pages 16 and 31.
- Robbins, H. and Monro, S. (1951). A stochastic approximation method. *The Annals of Mathematical Statistics*, 22(3):400–407. Cited on page 34.
- Roger, E. (1987). Stan Ulam, John Von Neumann, and the Monte Carlo method. *Los Alamos Science*, (15):131–137. Cited on page 47.
- Sabes, P. N. and Jordan, M. (1995). Reinforcement learning by probability matching. In *Advances in Neural Information Processing Systems*. Cited on page 31.
- Särkkä, S. and García-Fernández, Á. F. (2021). Temporal parallelization of Bayesian smoothers. *IEEE Transactions on Automatic Control*, 66(1):299–306. Cited on pages 58, 60, and 61.
- Särkkä, S. and Solin, A. (2019). *Applied Stochastic Differential Equations*. Cambridge University Press. Cited on page 57.
- Särkkä, S. and Svensson, L. (2023). *Bayesian Filtering and Smoothing*. Cambridge University Press, 2nd edition. Cited on pages 20, 23, 24, and 25.
- Savage, L. J. (1954). *The Foundations of Statistics*. Wiley, New York. Cited on pages 15 and 19.
- Schiesser, W. E. (2012). *The Numerical Method of Lines: Integration of Partial Differential Equations*. Elsevier. Cited on pages 58 and 66.
- Shannon, C. E. (1948). A mathematical theory of communication. *The Bell System Technical Journal*, 27(3):379–423. Cited on pages 40 and 42.
- Smallwood, R. D. and Sondik, E. J. (1973). The optimal control of partially observable Markov processes over a finite horizon. *Operations Research*, 21(5):1071–1088. Cited on page 37.
- Sondik, E. J. (1971). *The Optimal Control of Partially Observable Markov Processes*. Stanford University. Cited on pages 36 and 37.
- Stigler, S. M. (2007). The epic story of maximum likelihood. *Statistical Science*, 22(4):598–620. Cited on page 19.
- Suddath, J. H., Kidd III, R. H., and Reinhold, A. G. (1967). A linearized error analysis of onboard primary navigation systems for the Apollo lunar module. Technical report, National Aeronautics and Space Administration. Cited on page 15.
- Suomela, J. (2025). Programming parallel computers. <https://ppc.cs.aalto.fi/ch1/>. Accessed: 2025-11-07. Cited on page 58.
- Sutton, R. S. and Barto, A. G. (2018). *Reinforcement Learning: An Introduction*. MIT Press, 2nd edition. Cited on pages 30, 31, and 64.
- Thrun, S. (1999). Monte Carlo POMDPs. In *Advances in Neural Information Processing Systems*. Cited on page 65.
- Todorov, E. (2008). General duality between optimal control and estimation. In *Conference on Decision and Control*, pages 4286–4292. Cited on page 31.
- Toussaint, M. (2009). Robot trajectory optimization using approximate inference. In *International Conference on Machine Learning*. Cited on pages 29 and 31.

- Toussaint, M. and Storkey, A. (2006). Probabilistic inference for solving discrete and continuous state Markov decision processes. In *International Conference on Machine Learning*. Cited on pages 16 and 31.
- Tronarp, F., Särkkä, S., and Hennig, P. (2021). Bayesian ODE solvers: The maximum a posteriori estimate. *Statistics and Computing*, 31(23). Cited on page 56.
- Van Den Berg, J., Patil, S., and Alterovitz, R. (2012). Efficient approximate value iteration for continuous Gaussian POMDPs. In *AAAI Conference on Artificial Intelligence*, volume 26, pages 1832–1838. Cited on page 65.
- Vincent, B. T. (2016). Hierarchical Bayesian estimation and hypothesis testing for delay discounting tasks. *Behavior Research Methods*, 48:1608–1620. Cited on page 15.
- von Neumann, J. and Morgenstern, O. (1944). *Theory of Games and Economic Behavior*. Princeton University Press. Cited on page 40.
- Wainwright, M. J. and Jordan, M. I. (2008). Graphical models, exponential families, and variational inference. *Foundations and Trends® in Machine Learning*, 1(1-2):1–305. Cited on page 33.
- Williams, D. (1991). *Probability with Martingales*. Cambridge University Press. Cited on page 56.
- Yaghoobi, F., Corenflos, A., Hassan, S., and Särkkä, S. (2021). Parallel iterated extended and sigma-point Kalman smoothers. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 5350–5354. Cited on page 61.
- Yaghoobi, F., Corenflos, A., Hassan, S., and Särkkä, S. (2025). Parallel square-root statistical linear regression for inference in nonlinear state space models. *SIAM Journal on Scientific Computing*, 47(2):B454–B476. Cited on page 61.
- Zhang, D., Courville, A., Bengio, Y., Zheng, Q., Zhang, A., and Chen, R. T. Q. (2023). Latent state marginalization as a low-cost approach for improving exploration. In *International Conference on Learning Representations*. Cited on page 37.
- Ziebart, B. D., Maas, A., Bagnell, J. A., and Dey, A. K. (2008). Maximum entropy inverse reinforcement learning. In *Conference on Artificial Intelligence (AAAI)*. Cited on page 33.



Business, Economy  
Art, Design, Architecture  
Science, Technology  
Crossover

**| Doctoral Theses**

**Aalto DT 106/2026**

ISBN 978-952-64-3135-2  
ISBN 978-952-64-3134-5 (pdf)

**Aalto University**  
School of Electrical Engineering  
Department of Electrical Engineering  
and Automation  
**aalto.fi**