

Texts in Applied Mathematics 62

Kody Law · Andrew Stuart
Konstantinos Zygalakis

Data Assimilation

A Mathematical Introduction

EXTRAS ONLINE

 Springer

Texts in Applied Mathematics

Volume 62

Editors-in-chief:

Stuart Antman, University of Maryland, College Park, USA
Leslie Greengard, New York University, New York City, USA
Philip Holmes, Princeton University, Princeton, USA

Series Editors:

John B. Bell, Lawrence Berkeley National Lab, Berkeley, USA
Joseph B. Keller, Stanford University, Stanford, USA
Robert Kohn, New York University, New York City, USA
Paul Newton, University of Southern California, Los Angeles, USA
Charles Peskin, New York University, New York City, USA
Robert Pego, Carnegie Mellon University, Pittsburgh, USA
Lenya Ryzhik, Stanford University, Stanford, USA
Amit Singer, Princeton University, Princeton, USA
Angela Stevens, Universität Münster, Münster, Germany
Andrew Stuart, University of Warwick, Coventry, UK
Thomas Witelski, Duke University, Durham, USA
Stephen Wright, University of Wisconsin-Madison, Madison, USA

Kody Law • Andrew Stuart • Konstantinos Zygalakis

Data Assimilation

A Mathematical Introduction

Kody Law
Oak Ridge National Laboratory
Computer Science and Mathematics Division
Oak Ridge, USA

Andrew Stuart
Mathematics Institute
University of Warwick
Coventry, UK

Konstantinos Zygalakis
Division of Mathematical Sciences
University of Southampton
Southampton, UK

ISSN 0939-2475
Texts in Applied Mathematics
ISBN 978-3-319-20324-9
DOI 10.1007/978-3-319-20325-6

ISSN 2196-9949 (electronic)
ISBN 978-3-319-20325-6 (eBook)

Library of Congress Control Number: 2015945224

Mathematics Subject Classification (2010): 34, 35, 37, 60, 62, 65

Springer Cham Heidelberg New York Dordrecht London
© Springer International Publishing Switzerland 2015

This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

The publisher, the authors and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, express or implied, with respect to the material contained herein or for any errors or omissions that may have been made.

Printed on acid-free paper

Springer International Publishing AG Switzerland is part of Springer Science+Business Media (www.springer.com)

I dedicate this book to my wife, Natasha, to my parents, John and Julie, and to my sister, Amity.

KJHL.

For my step-children: Elliot, Gregory, and Zoe.

AMS.

Για την Γεωργία και το μικρό μας καραμελάκι, Χριστίνα.

Κώστας.

Contents

Preface	xii
List of Symbols	xvi
1 Mathematical Background	1
1.1 Probability	1
1.1.1 Random Variables on \mathbb{R}^{ℓ}	1
1.1.2 Gaussian Random Variables	3
1.1.3 Conditional and Marginal Distributions	6
1.1.4 Bayes's Formula	7
1.1.5 Independence	8
1.2 Dynamical Systems	9
1.2.1 Iterated Maps	9
1.2.2 Differential Equations	10
1.2.3 Long-Time Behavior	12
1.2.4 Controlled Dynamical Systems	14
1.3 Probability Metrics	15
1.3.1 Metric Properties	15
1.3.2 Metrics on Spaces of Probability Measures	15
1.4 Probabilistic View of Dynamical Systems	18
1.4.1 Markov Kernel	19
1.4.2 Ergodicity	20
1.4.3 Bayes's Formula as a Map	21
1.5 Bibliography	22
1.6 Exercises	22
2 Discrete Time: Formulation	25
2.1 Setup	25
2.2 Guiding Examples	26
2.3 Smoothing Problem	33
2.3.1 Probabilistic Formulation of Data Assimilation	33
2.3.2 Stochastic Dynamics	34
2.3.3 Reformulation of Stochastic Dynamics	36
2.3.4 Deterministic Dynamics	37
2.4 Filtering Problem	38
2.5 Filtering and Smoothing are Related	39
2.6 Well-Posedness	40
2.7 Assessing the Quality of Data-Assimilation Algorithms	43

2.8	Illustrations	45
2.9	Bibliographic Notes	49
2.10	Exercises	52
3	Discrete Time: Smoothing Algorithms	53
3.1	Linear Gaussian Problems: The Kalman Smoother	54
3.2	Markov Chain–Monte Carlo Methods	57
3.2.1	The MCMC Methodology	57
3.2.2	Metropolis–Hastings Methods	58
3.2.3	Deterministic Dynamics	60
3.2.4	Stochastic Dynamics	61
3.3	Variational Methods	65
3.4	Illustrations	68
3.5	Bibliographic Notes	74
3.6	Exercises	76
4	Discrete Time: Filtering Algorithms	79
4.1	Linear Gaussian Problems: The Kalman Filter	80
4.2	Approximate Gaussian Filters	82
4.2.1	3DVAR	84
4.2.2	Extended Kalman Filter	84
4.2.3	Ensemble Kalman Filter	84
4.2.4	Ensemble Square-Root Kalman Filter	86
4.3	The Particle Filter	87
4.3.1	The Basic Approximation Scheme	87
4.3.2	Sequential Importance Resampling	88
4.3.3	Improved Proposals	94
4.4	Large-Time Behavior of Filters	95
4.4.1	The Kalman Filter in One Dimension	95
4.4.2	The 3DVAR Filter	99
4.4.3	The Synchronization Filter	101
4.5	Illustrations	102
4.6	Bibliographic Notes	105
4.7	Exercises	113
5	Discrete Time: MATLAB Programs	115
5.1	Chapter 2 Programs	115
5.1.1	p1.m	116
5.1.2	p2.m	117
5.2	Chapter 3 Programs	119
5.2.1	p3.m	119
5.2.2	p4.m	122
5.2.3	p5.m	124
5.2.4	p6.m	126
5.2.5	p7.m	128
5.3	Chapter 4 Programs	130
5.3.1	p8.m	130
5.3.2	p9.m	133
5.3.3	p10.m	135
5.3.4	p11.m	137

5.3.5	p12.m	139
5.3.6	p13.m	141
5.3.7	p14.m	143
5.3.8	p15.m	145
5.4	ODE Programs	147
5.4.1	p16.m	147
5.4.2	p17.m	147
6	Continuous Time: Formulation	151
6.1	Setup	151
6.1.1	Derivation of the Continuous-Time Model	151
6.1.2	Stochastic Integration	153
6.2	Guiding Examples	156
6.3	Smoothing Problem	161
6.3.1	Probabilistic Formulation of Data Assimilation	161
6.3.2	Girsanov Formula	161
6.3.3	Conditional Probabilities	162
6.3.4	Stochastic Dynamics	163
6.3.5	Deterministic Dynamics	165
6.4	Filtering Problem	166
6.5	Illustrations	168
6.6	Bibliographic Notes	171
6.7	Exercises	173
7	Continuous Time: Smoothing Algorithms	175
7.1	Linear Gaussian Problems: The Kalman–Bucy Smoother	175
7.2	Markov Chain–Monte Carlo Methods	177
7.2.1	Deterministic Dynamics	177
7.2.2	Stochastic Dynamics	177
7.3	Variational Methods	179
7.4	Illustrations	180
7.5	Bibliographic Notes	184
7.6	Exercises	185
8	Continuous Time: Filtering Algorithms	187
8.1	Linear Gaussian Problems: The Kalman–Bucy Filter	188
8.2	Approximate Gaussian Filters	189
8.2.1	3DVAR	189
8.2.2	Extended Kalman Filter	190
8.2.3	Ensemble Kalman Filter	191
8.3	The Particle Filter	194
8.4	Large-Time Behavior of Filters	194
8.4.1	The Kalman–Bucy Filter in One Dimension	194
8.4.2	The 3DVAR Filter	195
8.5	Illustrations	197
8.6	Bibliographic Notes	198
8.7	Exercises	201

9 Continuous Time: MATLAB Programs	207
9.1 Chapter 6 Programs	207
9.1.1 p1c.m	207
9.1.2 p2c.m	208
9.1.3 p3c.m	209
9.2 Chapter 7 Programs	210
9.2.1 p4c.m	211
9.2.2 p5c.m	213
9.2.3 p6c.m	215
9.2.4 p7c.m	217
9.3 Chapter 8 Programs	219
9.3.1 p8c.m	219
9.3.2 p10c.m	221
9.3.3 p11c.m	222
9.3.4 p12c.m	222
9.3.5 p13c.m	225
9.3.6 p14c.m	225
9.3.7 p15c.m	225
9.3.8 p16c.m	229
9.3.9 p17c.m	229
References	233
Index	239

Preface

A central research challenge for the mathematical sciences in the twenty-first century is the development of principled methodologies for the seamless integration of (often vast) data sets with sophisticated mathematical models. Such data sets are becoming routinely available in almost all areas of engineering, science, and technology, while mathematical models describing phenomena of interest are often built on decades, or even centuries, of human knowledge. Ignoring either the data or the models is clearly unwise, and so the issue of combining them is of paramount importance. When the underlying mathematical model is a (possibly stochastic) dynamical system and the data may be time-ordered, combining model and data is referred to as **data assimilation**.

The research area of data assimilation has been driven, to a large extent, by practitioners working in the atmospheric and oceanographic sciences and in other areas of the geosciences, such as oil recovery. The resulting research has led to a host of algorithmic approaches and a number of significant algorithmic innovations. However, there has been no systematic treatment of the mathematical underpinnings of the subject. The goal of this book is to provide such a treatment. Specifically, we develop a unified mathematical framework in which a Bayesian formulation of the problem provides the bedrock for the derivation and development of algorithms; furthermore, the examples used in the text, together with the algorithms that are introduced and discussed, are all illustrated by MATLAB software detailed in the book and freely available online via the Springer website, the authors' personal web pages, and at the following link:

<http://tiny.cc/damat> .

It is important to appreciate that this book develops the subject of data assimilation in a manner that differs significantly from both its historical development and its typical presentation in other books. We begin with a “gold-standard” Bayesian formulation of the problem, which while out of reach for current online geophysical applications such as weather forecasting, provides a clearly defined mathematical problem whose solution would provide the ideal combination of mathematical model and available data, given known statistical uncertainties in both. We then describe various algorithmic approaches within the context of the underpinning Bayesian formulation. The reader interested in understanding the historical development of data assimilation in an applied context, or in more practically oriented mathematical and computational treatments of the field, has a number of options for additional reading, which we now describe. First, we mention that the probabilistic “smoothing” and “filtering” approaches that we describe in this book grew out of the calculus of variations and the control-theoretic viewpoints on data assimilation, resulting, respectively, from the papers of Talagrand and Courtier [135, 37] and Kalman [79]. The current practice of data assimilation in the context of the atmospheric sciences, and weather prediction in particular, is covered in the book by Kalnay [81]. An introduction in which the presentation is motivated by applications in oceanography is Bennett’s book [12]. The book by Evensen [50] provides

a good overview of many computational aspects of the subject, reflecting the author's experience in geophysical applications including oceanography and oil recovery; application of data assimilation in oil recovery is the primary focus of the book [115] by Oliver, Reynolds, and Li. The recent book by Abarbanel provides a physics and dynamical systems perspective on data assimilation [1], with motivation coming not only from weather forecasting, but also from neuroscience. The reader interested in state-of-the-art novel online algorithms based on mathematical insight and physical reasoning may consult the book by Majda and Harlim [100]; there the focus is on complex dynamical systems exhibiting turbulent behavior. And finally, the book by Cotter and Reich [31] provides a novel numerical-analysis-based perspective on the field, with primary motivation coming from geophysical applications. Our book provides a mathematical perspective on the subject via which all of these existing books can be interpreted.

The book is organized into nine chapters: the first contains a brief introduction to the mathematical tools around which the material is organized; the next four are concerned with discrete-time dynamical systems and discrete-time data, while the last four are concerned with continuous-time dynamical systems and continuous-time data; continuous-time dynamical systems together with discrete-time data can be reformulated as discrete-time dynamical systems with discrete-time data, and so this problem is not covered explicitly in the book. The four chapters on discrete and continuous time are organized identically: in the first, we frame the problem, the second and third are devoted to smoothing algorithms and filtering algorithms respectively, and the fourth contains MATLAB programs and discussion of them. Both underlying stochastic and deterministic dynamics are studied.

Chapter 1 is organized around four themes: probability, dynamical systems, probability metrics, and dynamical systems for probability measures. These subjects are given a terse overview that cannot do justice to the richness of these areas of mathematics. However, the introductory style of the material serves to orient the reader toward the form of mathematical thinking that underpins the approach taken to data assimilation in this book. Chapter 2 sets up the problem of data assimilation, based on an underlying stochastic dynamical system in which the noise appears additively; the degenerate case in which the noise disappears, deterministic dynamics, is also considered. Nonadditive noise can be studied similarly to our development, but it does lead to additional complications in some scenarios, in particular when the transition density of the underlying Markov chain is not known explicitly; for pedagogical reasons, we hence focus on additive noise. Similarly, the data is assumed to be found from a nonlinear function of the output of the dynamical system, at each discrete time, also with additive noise. Furthermore, both the model and data noise are assumed to be Gaussian in order to simplify our exposition; this may easily be relaxed to include situations in which the noise has a known density with respect to Lebesgue measure. Examples of illustrative maps are given, including a pair of examples arising from quadratic differential equations, both due to Lorenz, of dissipative character and motivated by geophysical applications. The central conceptual idea underlying the chapter is to frame the data-assimilation problem as that of determining the probability distribution on the output of the dynamical system when conditioned on the observed noisy data—the fully Bayesian formulation of the problem. There are two key variants on this perspective: smoothing, in which data in the future may condition knowledge of the output of the dynamical system in its past; and filtering, in which data in the future is not used to estimate the conditional probability distribution at a given time. We define these two probabilistic problems, demonstrate that they are related to one another, and describe a well-posedness theory, showing that the desired conditional probability distributions are Lipschitz with respect to small changes in the data.

The two probabilistic problems of smoothing and filtering lead to two different classes of algorithms, and these are described, respectively, in Chapters 3 and 4. Smoothing algorithms are

typically more computationally intensive than filtering algorithms. This is because smoothing algorithms involve the incorporation of data that is distributed over a time interval, while filtering algorithms incorporate the data sequentially, at each time point, as it arrives. Adapting the output of a dynamical system to fit a stream of data all at once, rather than sequentially, is typically a more demanding task; however, it does have potential rewards, and at least currently, weather forecasts based on the former approach give better predictions than those based on sequential incorporation of data. Chapter 3 starts with a fully Bayesian algorithmic approach, namely the use of Monte Carlo–Markov Chain (MCMC) methods to sample the distribution on the output of the dynamical system, given the data. These methods are introduced in a general framework for an arbitrary target density defined via its Lebesgue density, and an arbitrary proposal distribution; then a variety of specific methods are described, exploiting the specific structure of the target distribution arising in the data-assimilation problem to construct the proposal. The chapter concludes by linking the Bayesian approach to variational methods; specifically, it is shown how the well-known 4DVAR (for deterministic dynamics) and weak constraint 4DVAR (for stochastic dynamics, denoted by w4DVAR in this book) methods correspond to maximizing the desired probability distribution on signal given data. In Chapter 4, filtering algorithms are described. For the derivation of many of the algorithms, we adopt a minimization approach; this unifies all the methods described, with the exception of the particle filter. The chapter begins by studying linear Gaussian problems, where the Kalman filter may be used to characterize the resulting Gaussian probability distribution—the filtering distribution—exactly. The propagation of the mean is shown to be derived from a sequential minimization principle in which a quadratic functional, representing a compromise between fitting the model and the data, is minimized as each data point is acquired in time. This sequential minimization approach is then used as the basis from which to derive various approximate algorithms such as 3DVAR, the extended Kalman filter, and the ensemble Kalman filter. Although widely used in practice, these approximate algorithms cannot in general accurately reproduce the desired conditional probability distribution, except for linear Gaussian problems. Our final algorithm in this chapter is the particle filter, which, although known to behave poorly in high dimensions, can in principle reproduce the true filtering distribution as the number of particles tends to infinity. Neither the MCMC method for the smoothing problem nor the particle filter for the filtering problem is currently practical for high-dimensional applications such as those arising in the geophysical sciences. However, they play an important role in the subject, since they provide benchmarks against which the more practical, yet less well founded, algorithms may be compared. Furthermore, their formulation provides guiding principles that can be used in the development of new algorithms targeted at the high-dimensional setting.

In Chapter 2, the guiding dynamical system examples used throughout the text are illustrated with the output of MATLAB programs, while Chapters 3 and 4 include the output of MATLAB programs to illustrate the algorithms we introduce for smoothing and filtering. In Chapter 5, we provide and discuss the MATLAB programs used to produce all of this material. These programs are provided for two reasons: firstly, for some readers, code can provide a very direct way to understand both the theory and algorithms presented within the text; and secondly, the codes can form the basis of more sophisticated algorithms that the reader can develop by building upon them. We have not aimed for the most succinct or speediest implementation of the algorithms, but rather have tried to maintain a uniform presentation in which connections to the theoretical developments in the text are fairly transparent.

Chapter 6 introduces data assimilation in continuous time. Our perspective in this chapter, and in the two subsequent algorithmic chapters concerning continuous time, is to motivate much of the mathematics by taking formal asymptotic limits of the discrete-time setting, in which the underlying dynamical system behaves like a stochastic differential equation

(SDE), as does the observation model. Continuous-time data assimilation is in principle a very technical area, primarily because it involves probability on infinite-dimensional spaces. Our approach is to build intuition about the problems using the discrete setting as a stepping-stone, deriving continuous-time limits from increasingly frequent discrete-time observations of an underlying continuous process. Although we must forgo complete rigorous proofs in order to adopt this approach within a short book, we believe that the resulting intuition will help the reader access the more technical published literature in this area, and that the lack of rigorous proofs is compensated for by significant insight. Various important results from the theory of stochastic differential equations are stated and then used throughout the chapter, including the Itô and the Girsanov formulae, together with basic existence and uniqueness theorems for equations with additive Brownian noise and equipped with moment inequalities; such inequalities arise naturally for the dissipative quadratic dynamical systems introduced in Chapter 2. As in discrete time, the smoothing and filtering distributions are both introduced. Furthermore, the Zakai and Kushner–Stratonovich stochastic partial differential equations (SPDEs) for the evolution of the density of the filtering distribution are also derived.

In Chapter 7, we describe MCMC methods to sample the posterior distribution for the smoothing problem in continuous time. We include both stochastic dynamics, for which the probability distribution of interest is on an infinite-dimensional space (the pathspace of the solution on a time interval $[0, T]$), and deterministic dynamics, for which it is on a finite-dimensional space (where the initial condition lies). As in the discrete-time setting, we also discuss variational methods, which lead to problems in the calculus of variations. Chapter 8 is devoted to filtering algorithms in continuous time. The Kalman–Bucy filter is derived by taking the small interobservation time limit of a discretely observed continuous process. The same methodology is then applied to the 3DVAR method, to the extended and ensemble Kalman filters, and to the particle filter. In Chapter 9, we provide and discuss the MATLAB code used to produce the figures in the preceding three chapters; our motivation for including this code is the same as in Chapter 5, where we consider discrete-time data assimilation.

Warning We reiterate an important issue relating to the perspective we adopt in this book. Our aim is to provide clearly defined mathematical foundations for data assimilation, and this leads to the Bayesian problem of finding a probability distribution on the signal, given the data. In the very-high-dimensional problems arising in many applications, especially those of geophysical origin referred to above, it is currently beyond reach to expect to be able to compute the resulting probability distributions accurately in high dimensions, in either the filtering or the smoothing context. Thus many of the algorithms we describe are currently impractical for such applications, especially in online scenarios. For this reason, of all the algorithms we describe here, only the 3DVAR, 4DVAR, and ensemble Kalman filter are routinely used in real geophysical applications. The use of MCMC methods for smoothing problems, and the extended Kalman filter and the particle filter, is currently impractical for real online applications arising for large systems such as those in geophysical applications. However, this fact should not diminish their importance. Our book provides a clear and unequivocal statement and analysis of the desired problem to be solved in many such applications, and methods that can be used to accurately solve the problem in various simplified scenarios. As such, we believe that it provides an important cornerstone in the field that can help guide applied researchers.

Target Audience/Prerequisites The book is aimed at mathematical researchers interested in a systematic development of this interdisciplinary field and at researchers from the geosciences and a variety of other scientific fields who use tools from data assimilation to combine data with time-dependent models. As well as being suitable for self-study by such researchers,

the book can also be used as a text, and the examples, exercises, and MATLAB programs help to make it usable in this context. Although we review some basic mathematical prerequisites in the first chapter, a first course in each of differential equations, dynamical systems, and probability is assumed of the reader. It is thus suitable for a master's- or PhD-level lecture course.

Notation Throughout, $(\langle \cdot, \cdot \rangle, |\cdot|)$ denotes the Euclidean inner product and norm on \mathbb{R}^ℓ , for an integer ℓ ; and $|\cdot|$ will also denote the induced operator norm. In the sections on continuous time, the same notation will also be used for the Hilbert space structure on the space $L^2([0, T]; \mathbb{R}^\ell)$. For a positive definite symmetric matrix $A \in \mathbb{R}^{\ell \times \ell}$, we introduce the inner product $\langle \cdot, \cdot \rangle_A = \langle A^{-\frac{1}{2}} \cdot, A^{-\frac{1}{2}} \cdot \rangle$ and the resulting norm $|\cdot|_A = |A^{-\frac{1}{2}} \cdot|$; this may also be generalized to the Hilbert space setting of $L^2([0, T]; \mathbb{R}^\ell)$. We let $\|\cdot\|_F$ denote the Frobenius norm of a matrix, found as the square root of the sum of the squares of the entries. The outer product of the ordered pair of vectors a, b in \mathbb{R}^ℓ is the linear operator $a \otimes b$ with property $(a \otimes b)c = \langle b, c \rangle a$. The symbol \wedge is used to denote the (symmetric) operation on a pair of real numbers that delivers the minimum of that pair: $a \wedge b = a$ if $a \leq b$.

The symbol $\mathbb{N} = \{1, 2, \dots\}$ denotes the positive integers, and $\mathbb{Z}^+ := \mathbb{N} \cup \{0\} = \{0, 1, 2, \dots\}$. We write \mathbb{R} for $(-\infty, \infty)$ and \mathbb{R}^+ for $[0, \infty)$. We use > 0 (respectively ≥ 0) to denote positive definite (respectively positive semidefinite) for real symmetric matrices. The notation $\mathbb{R}_{\text{sym}}^{\ell \times \ell}$ is used to denote symmetric $\ell \times \ell$ matrices. The symbols \mathbb{P} and \mathbb{E} are used to denote probability (measure and density function) and expectation.

A Gaussian probability distribution on \mathbb{R}^ℓ , or on the Hilbert space $L^2([0, T]; \mathbb{R}^\ell)$, will be denoted by $N(m, C)$, with m the *mean* and *covariance* C . The inverse of the covariance, the *precision*, is denoted by the symbol L . The symbols σ and γ are typically used to denote standard deviations. The symbol cf denotes a characteristic function, and SI a stochastic integral. The symbol \mathbf{z} denotes an i.i.d. (independent and identically distributed) sequence of uniform random variables on $[0, 1]$, and $\boldsymbol{\iota}$ an i.i.d. sequence of Gaussian random variables, both arising in the definition of MCMC methods.

We use v to denote the (unknown) signal, and y the data, both indexed by discrete time j . The symbol Y_j denotes the accumulated data to time j . In continuous time, the unknown signal is again v , now indexed by continuous time t , and z denotes the data, also indexed by t . The maps Ψ and f on \mathbb{R}^n define the systematic part of the dynamics model in discrete and continuous time respectively, while h denotes the observation operator in both discrete and continuous time. The symbols ξ and η denote the noise sequences entering the signal and data models, respectively, in discrete time; when these random variables are Gaussian, their covariances are Σ and Γ respectively. In continuous time, noises are typically Brownian, denoted by B and W (possibly with suffixes) and variances Σ_0 and Γ_0 . The symbols \mathbf{v} and \mathbf{q} denote variables determined from v and h respectively.

The symbols $\mu, \mu_0, \mu', \mu_j, \vartheta, \vartheta_0, \nu$, some possibly with suffixes, denote probability measures. The symbols $\rho, \rho_0, \rho', \rho_j, \pi, \pi_0, \varrho$, again possibly with suffixes, denote probability density functions (pdfs). We use μ_∞ (respectively ρ_∞) to denote (typically ergodic) invariant measures (respectively densities). Furthermore, p denotes a Markov kernel. The symbol P is used to denote the linear map on measures (or their densities) implied by a Markov kernel, and L_j the nonlinear map on measures (or their densities) implied by application of Bayes's formula to take a prior into a posterior.

The symbols I and J , possibly with suffixes \det or r , are used to denote nonlinear real-valued functionals arising in variational methods, and the functionals Φ (in discrete time) and Ξ_1, Ξ_2 , and Ξ_3 (in continuous time) appear in their definition.

Acknowledgments Some of the research described in the book has emerged from collaborations with others, and the authors are indebted to all of their coauthors on collaborative work cited in the bibliography sections. The authors are very grateful to Sergios Agapiou, Daniel Sanz-Alonso, and Yuan-Xiang Zhang for help in typesetting parts of this book, and for input regarding the presentation; furthermore, Ian Hawke and Gregory Ashton provided help with the preparation of the figures, while Daniel Sanz-Alonso road-tested many of the exercises. The authors also wish to thank Mel Ades and Håkon Hoel for useful feedback, which helped to improve the presentation. The anonymous reviewers chosen by Springer and editors at Springer also gave very useful feedback, and Achi Dosanjh from Springer provided very helpful overall guidance and support throughout the publication process. And finally, the authors thank the student audiences at Peking University (2012), KAUST (2014), the University of Warwick (2014), and Fudan University (2015), whose interest helped to shape the notes. AMS is grateful to EPSRC, ERC, ESA, and ONR for financial support for research whose development suggested the need for a mathematization of the subject of data assimilation. KJHL was supported as a member of the SRI-UQ Center at KAUST while much of the writing of this book was undertaken.

It is hard to perfect a book. The authors would be grateful to receive any comments and suggestions for improvement, by email:

a.m.stuart@warwick.ac.uk .

Please note that in making a bibliography for this book, we have not tried to be comprehensive, but rather to ensure that we have included key references relating to the mathematical underpinnings of data assimilation that are our focus here, together with texts and other references that support this mathematical perspective.

Oak Ridge, USA
Coventry, UK

Southampton, UK
April 2015

K.J.H. Law
A.M. Stuart
K.C. Zygalakis

List of symbols

Symbol	Description
$(\langle \cdot, \cdot \rangle, \cdot)$	Euclidean inner product and norm on \mathbb{R}^ℓ , for any integer ℓ ; occasionally extended to the Hilbert space $L^2([0, T]; \mathbb{R}^\ell)$
$\langle \cdot, \cdot \rangle_A = \langle A^{-\frac{1}{2}} \cdot, A^{-\frac{1}{2}} \cdot \rangle$	weighted inner product (A positive definite symmetric)
$ \cdot _A = A^{-\frac{1}{2}} \cdot $	A induced norm (A positive definite symmetric)
$\ \cdot\ _F$	Frobenius norm of a matrix F
\mathbb{N}	$\{1, 2, \dots\}$
\mathbb{Z}^+	$\{0, 1, 2, \dots\}$
$a \otimes b$	outer product of the ordered pair of vectors a, b in \mathbb{R}^ℓ (with property $(a \otimes b)c = \langle b, c \rangle a$)
\mathbb{R}	$(-\infty, \infty)$
\mathbb{R}^+	$[0, \infty).$
$A > 0$	A positive definite real symmetric matrix
$A \geq 0$	A positive semi-definite real symmetric matrix
$\mathbb{R}^{\ell \times \ell}_{\text{sym}}$	set of symmetric $\ell \times \ell$ matrices
\mathbb{P}, \mathbb{E}	probability (measure or density) and expectation
$N(m, C)$	Gaussian probability distribution with mean m and covariance C on \mathbb{R}^ℓ or on the Hilbert space $L^2([0, T]; \mathbb{R}^\ell)$
L	inverse of the covariance matrix or operator (also called precision)
σ, γ	standard deviations
cf	characteristic function
SI	stochastic integral
r	i.i.d. sequence of uniform random variables on $[0, 1]$ arising in the definition of MCMC methods
ι	i.i.d. sequence of Gaussian random variables arising in the definition of MCMC methods
v	unknown signal (indexed by discrete time j)
y	data (indexed by discrete time j)
Y_j	accumulated data up to time j : $\{y_\ell\}_{\ell=1}^j$
v	unknown signal (indexed by continuous time t)
z	data (indexed by continuous time t)
Ψ	systematic part of dynamics (discrete time)
f	systematic part of dynamics (continuous time)
ξ	noise sequence entering the signal model (discrete time)
η	noise sequence entering the data model (discrete model)
Σ, Γ	covariance of ξ and η
B	Brownian motion entering the signal model (continuous time)
W	Brownian motion entering the data model (continuous time)
Σ_0, Γ_0	covariance of B and W respectively.
v	variable determined from v (used to prove well-posedness)
q	variable determined from h (used in derivation of Zakai equation)
$\mu, \mu_0, \mu', \mu_j, \vartheta, \vartheta_0, \nu$	probability measures
$\rho, \rho_0, \rho', \rho_j, \pi, \pi_0, \varrho$	probability density functions (associated with those measures)
ρ_∞	ergodic invariant (probability) measure
ρ_∞	ergodic invariant (probability) density function.

p	Markov kernel
P	linear map on measures (or densities) implied by a Markov kernel
L_j	nonlinear map on measures (or their densities) implied by application of Bayes's formula to take a prior into a posterior
I and J	real-valued functionals arising in variational methods; possibly with suffices denoting deterministic or reformulated problems
Φ, Ξ_1, Ξ_2 and Ξ_3	appear in definition of these functionals arising in variational methods

Chapter 1

Mathematical Background

The purpose of this chapter is to provide a brief overview of the key mathematical ways of thinking that underpin our presentation of the subject of data assimilation. In particular, we touch on the subjects of probability, dynamical systems, probability metrics, and dynamical systems for probability measures, in Sections 1.1, 1.2, 1.3, and 1.4 respectively. Our treatment is necessarily terse and very selective, and the bibliography section 1.5 provides references to the literature. We conclude with exercises in Section 1.6.

We highlight here the fact that throughout this book, all probability measures on \mathbb{R}^ℓ will be assumed to possess a density with respect to Lebesgue measure, and furthermore, this density will be assumed to be strictly positive everywhere in \mathbb{R}^ℓ . This assumption simplifies greatly our subsequent probabilistic calculations.

1.1 Probability

We describe here some basic notation and facts from probability theory, all of which will be fundamental to formulating data assimilation from a probabilistic perspective.

1.1.1. Random Variables on \mathbb{R}^ℓ

We consider a *random variable* z , defined as a function on an underlying probability space and taking values in \mathbb{R}^ℓ . Associated with this random variable is an induced probability measure μ on \mathbb{R}^ℓ . Furthermore, to be able to compute expectations, we need to work with a sufficiently rich collection of subsets of \mathbb{R}^ℓ , to each of which we can assign the probability that z is contained in it; this collection of subsets is termed a *σ -algebra*. Throughout these notes we work with $\mathcal{B}(\mathbb{R}^\ell)$, the Borel σ -algebra generated by the open sets; we will abbreviate this σ -algebra by \mathcal{B} when the set \mathbb{R}^ℓ is clear. The Borel σ -algebra is the natural collection of subsets available on \mathbb{R}^ℓ that allows for coherent assignation of probabilities and a theory of integration; an element in \mathcal{B} will be termed a *Borel set*.

We have defined a *probability triple* $(\mathbb{R}^\ell, \mathcal{B}, \mu)$. For simplicity, we assume throughout the book that z has a strictly positive *probability density function* (pdf) ρ with respect to Lebesgue

measure.¹ Then for every Borel set $A \subset \mathbb{R}^\ell$, we define $\mu(A)$, the probability that the random variable z lies in A , by

$$\mu(A) = \int_A \rho(x) dx,$$

where the pdf $\rho : \mathbb{R}^\ell \rightarrow \mathbb{R}^+$ satisfies

$$\int_{\mathbb{R}^\ell} \rho(x) dx = 1.$$

A Borel set $A \subset \mathbb{R}^\ell$ is sometimes termed an *event*, and the event is said to occur *almost surely* if $\mu(A) = 1$. Since ρ integrates to 1 over \mathbb{R}^ℓ and is strictly positive, this implies that the Lebesgue measure of the complement of A , the set A^c , is zero.

We write $z \sim \mu$ as shorthand for the statement that z is *distributed* according to probability measure μ on \mathbb{R}^ℓ . Note that here $\mu : \mathcal{B}(\mathbb{R}^\ell) \rightarrow [0, 1]$ denotes a probability measure, and $\rho : \mathbb{R}^\ell \rightarrow \mathbb{R}^+$ the corresponding density. However, we will sometimes use the letter \mathbb{P} to denote both the measure and its corresponding pdf. This should create no confusion: $\mathbb{P}(\cdot)$ will be a probability measure whenever its argument is a Borel set, and a density whenever its argument is a point in \mathbb{R}^ℓ . On occasion, we will write $\mathbb{P}(z \in A)$ for $\mu(A)$.

For a function $f : \mathbb{R}^\ell \rightarrow \mathbb{R}^{p \times q}$, we denote by $\mathbb{E}f(z)$ the *expected value* of the random variable $f(z)$ on $\mathbb{R}^{p \times q}$; this expectation is given by

$$\mathbb{E}f(z) = \int_{\mathbb{R}^\ell} f(x)\mu(dx), \quad \mu(dx) = \rho(x)dx.$$

We also sometimes write $\mu(f)$ for $\mathbb{E}f(z)$. The case in which the function f is vector-valued corresponds to $q = 1$, so that $\mathbb{R}^{p \times q} = \mathbb{R}^{p \times 1} \equiv \mathbb{R}^p$. We will sometimes write \mathbb{E}^μ if we wish to differentiate between different measures with respect to which the expectation is to be understood.

The *characteristic function* of the random variable z on \mathbb{R}^ℓ is $\text{cf} : \mathbb{R}^\ell \rightarrow \mathbb{C}$, defined by

$$\text{cf}(h) = \mathbb{E} \exp(i\langle h, z \rangle).$$

Example 1.1. Let $\ell = 1$, and set $\rho(x) = \frac{1}{\pi(1+x^2)}$. Note that $\rho(x) > 0$ for every $x \in \mathbb{R}$. Also, using the change of variables $x = \tan \theta$, we have

$$\int_{-\infty}^{\infty} \frac{dx}{\pi(1+x^2)} = 2 \int_0^{\infty} \frac{dx}{\pi(1+x^2)} = \int_{\arctan(0)}^{\arctan(\infty)} \frac{2 \sec^2 \theta d\theta}{\pi(1+\tan^2 \theta)} = \frac{2}{\pi} \int_0^{\pi/2} d\theta = 1,$$

and therefore ρ is the pdf of a random variable z on \mathbb{R} . We say that such a random variable has the *Cauchy distribution*. ♠

Let $G : \mathbb{R}^\ell \rightarrow \mathbb{R}^\ell$, and note that if z is a random variable on \mathbb{R}^ℓ , then so too is $G(z)$. If $z \sim \mu$ then $G(z) \sim G \star \mu$, the *pushforward* of μ under G . If μ has associated pdf ρ on \mathbb{R}^ℓ , then $G \star \mu$ has associated pushforward pdf, denoted by $G \star \rho$. This pushforward pdf may be calculated explicitly by means of the change of variable formula under an integral. Indeed if G is invertible, then

$$G \star \rho(v) := \rho(G^{-1}(v))|DG^{-1}(v)|.$$

¹ Of course, μ and ρ depend on the particular random variable z , but we suppress this dependence in the notation.

We will occasionally use the *Markov inequality*, which states that for a random variable z on \mathbb{R}^ℓ and $R > 0$,

$$\mathbb{P}(|z| \geq R) \leq R^{-1}\mathbb{E}|z|. \quad (1.1)$$

As a consequence,

$$\mathbb{P}(|z| < R) \geq 1 - R^{-1}\mathbb{E}|z|. \quad (1.2)$$

In particular, if $\mathbb{E}|z| < \infty$, then choosing R sufficiently large shows that $\mathbb{P}(|z| < R) > 0$. In our setting, this last inequality follows in any case, by assumption of the strict positivity of $\rho(\cdot)$ everywhere on \mathbb{R}^ℓ .

A sequence of probability measures $\mu^{(n)}$ on \mathbb{R}^ℓ is said to *converge weakly* to a limiting probability measure μ on \mathbb{R}^ℓ if for all continuous bounded functions $\varphi : \mathbb{R}^\ell \rightarrow \mathbb{R}$, we have

$$\mathbb{E}^{\mu^{(n)}}\varphi(u) \rightarrow \mathbb{E}^\mu\varphi(u)$$

as $n \rightarrow \infty$.

Finally, we note that although developed here on \mathbb{R}^ℓ , the theory of probability can be developed on much more general measure spaces and, in particular, on separable Banach spaces. In the part of the book relating to continuous time, we will use probability theory in this setting.

1.1.2. Gaussian Random Variables

We work in finite dimensions, but all the ideas can be generalized to infinite-dimensional contexts, such as the separable Hilbert space setting. A *Gaussian* random variable² on \mathbb{R}^ℓ is characterized by the following parameters:

- Mean: $m \in \mathbb{R}^\ell$.
- Covariance: $C \in \mathbb{R}_{\text{sym}}^{\ell \times \ell}$, $C \geq 0$.

We write $z \sim N(m, C)$ and call the Gaussian random variable *centered* if $m = 0$. If $C > 0$, then z has strictly positive pdf on \mathbb{R}^ℓ , given by

$$\rho(x) = \frac{1}{(2\pi)^{\ell/2}(\det C)^{1/2}} \exp\left(-\frac{1}{2}|C^{-\frac{1}{2}}(x - m)|^2\right) \quad (1.3a)$$

$$= \frac{1}{(2\pi)^{\ell/2}(\det C)^{1/2}} \exp\left(-\frac{1}{2}|x - m|_C^2\right). \quad (1.3b)$$

It can be shown that indeed ρ given by (1.3) satisfies

$$\int_{\mathbb{R}^\ell} \rho(x) dx = 1. \quad (1.4)$$

Lemma 1.2. *Let $z \sim N(m, C)$, $C > 0$. Then*

1. $\mathbb{E}z = m$.
2. $\mathbb{E}(z - m)(z - m)^T = C$.

² Sometimes also called a *normal* random variable.

Proof For the first item,

$$\begin{aligned}
\mathbb{E}z &= \frac{1}{(2\pi)^{\ell/2}(\det C)^{1/2}} \int_{\mathbb{R}^\ell} x \exp\left(-\frac{1}{2}|x - m|_C^2\right) dx \\
&= \frac{1}{(2\pi)^{\ell/2}(\det C)^{1/2}} \int_{\mathbb{R}^\ell} (y + m) \exp\left(-\frac{1}{2}|y|_C^2\right) dy \\
&= \frac{1}{(2\pi)^{\ell/2}(\det C)^{1/2}} \int_{\mathbb{R}^\ell} y \exp\left(-\frac{1}{2}|y|_C^2\right) dy + \frac{m}{(2\pi)^{\ell/2}(\det C)^{1/2}} \int_{\mathbb{R}^\ell} \exp\left(-\frac{1}{2}|y|_C^2\right) dy \\
&= 0 + m \\
&= m,
\end{aligned}$$

where we used in the last line that the function $y \mapsto y \exp\left(-\frac{1}{2}|y|_C^2\right)$ is even and the fact that by (1.4),

$$\frac{1}{(2\pi)^{\ell/2}(\det C)^{1/2}} \int_{\mathbb{R}^\ell} \exp\left(-\frac{1}{2}|y|_C^2\right) dy = 1.$$

For the second item,

$$\begin{aligned}
\mathbb{E}(z - m)(z - m)^T &= \frac{1}{(2\pi)^{\ell/2}(\det C)^{1/2}} \int_{\mathbb{R}^\ell} (x - m)(x - m)^T \exp\left(-\frac{1}{2}|x - m|_C^2\right) dx \\
&= \frac{1}{(2\pi)^{\ell/2}(\det C)^{1/2}} \int_{\mathbb{R}^\ell} yy^T \exp\left(-\frac{1}{2}|C^{-1/2}y|^2\right) dy \\
&= \frac{1}{(2\pi)^{\ell/2}(\det C)^{1/2}} \int_{\mathbb{R}^\ell} C^{1/2}ww^TC^{1/2} \exp\left(-\frac{1}{2}|w|^2\right) \det(C^{1/2}) dw \\
&= C^{1/2}JC^{1/2},
\end{aligned}$$

where

$$J = \frac{1}{(2\pi)^{\ell/2}} \int_{\mathbb{R}^\ell} ww^T \exp\left(-\frac{1}{2}|w|^2\right) dw \in \mathbb{R}^\ell \times \mathbb{R}^\ell,$$

and so

$$J_{ij} = \frac{1}{(2\pi)^{\ell/2}} \int_{\mathbb{R}^\ell} w_i w_j \exp\left(-\frac{1}{2} \sum_{k=1}^{\ell} w_k^2\right) \prod_{k=1}^{\ell} dw_k.$$

To complete the proof, we need to show that J is the identity matrix I on $\mathbb{R}^\ell \times \mathbb{R}^\ell$. Indeed, for $i \neq j$,

$$J_{ij} \propto \int_{\mathbb{R}} w_i \exp\left(-\frac{1}{2}w_i^2\right) dw_i \int_{\mathbb{R}} w_j \exp\left(-\frac{1}{2}w_j^2\right) dw_j = 0,$$

by symmetry; and for $i = j$,

$$\begin{aligned}
J_{jj} &= \frac{1}{(2\pi)^{\frac{1}{2}}} \int_{\mathbb{R}} w_j^2 \exp\left(-\frac{1}{2}w_j^2\right) dw_j \left(\frac{1}{(2\pi)^{\frac{1}{2}}} \int_{\mathbb{R}} \exp\left(-\frac{1}{2}w_k^2\right) dw_k \right)^{\ell-1} \\
&= \frac{1}{(2\pi)^{\frac{1}{2}}} \int_{\mathbb{R}} w_j^2 \exp\left(-\frac{1}{2}w_j^2\right) dw_j \\
&= -\frac{1}{(2\pi)^{\frac{1}{2}}} w_j \exp\left(-\frac{1}{2}w_j^2\right) \Big|_{-\infty}^{\infty} + \frac{1}{(2\pi)^{\frac{1}{2}}} \int_{\mathbb{R}} \exp\left(-\frac{1}{2}w_j^2\right) dw_j = 1,
\end{aligned}$$

where we again used (1.4) in the first and last lines. Thus $J = I$, the identity in \mathbb{R}^ℓ , and $\mathbb{E}(z - m)(z - m)^T = C^{1/2}C^{1/2} = C$. \square

The following characterization of Gaussians is often useful.

Lemma 1.3. *The characteristic function of the Gaussian $N(m, C)$ is given by*

$$\text{cf}(h) = \exp\left(i\langle h, m \rangle - \frac{1}{2}\langle Ch, h \rangle\right).$$

Proof This follows from noting that

$$\frac{1}{2}|x - m|_C^2 - i\langle h, x \rangle = \frac{1}{2}|x - (m + iCh)|_C^2 - i\langle h, m \rangle + \frac{1}{2}\langle Ch, h \rangle.$$

\square

Remark 1.4. Note that the pdf for the Gaussian random variable that we wrote down in equation (1.3) is defined only for $C > 0$, since it involves C^{-1} . The characteristic function appearing in the preceding lemma can be used to define a Gaussian with mean m and covariance C , including the case $C \geq 0$, so that the Gaussian covariance C is only positive semidefinite, since it is defined in terms of C and not C^{-1} . For example, if we let $z \sim N(m, C)$ with $C = 0$, then z is a Dirac mass at m , i.e., $z = m$ almost surely, and for every continuous function f ,

$$\mathbb{E}f(z) = f(m).$$

This Dirac mass may be viewed as a particular case of a Gaussian random variable. We will write δ_m for $N(m, 0)$. \spadesuit

Lemma 1.5. *The following hold for Gaussian random variables:*

- If $z = a_1 z_1 + a_2 z_2$, where z_1, z_2 are independent Gaussians with distributions $N(m_1, C_1)$ and $N(m_2, C_2)$ respectively, then z is Gaussian with distribution $N(a_1 m_1 + a_2 m_2, a_1^2 C_1 + a_2^2 C_2)$.
- If $z \sim N(m, C)$ and $w = Lz + a$, then $w \sim N(Lm + a, LCL^T)$.

Proof The first result follows from computing the characteristic function of z . By independence, this is the product of the characteristic functions of $a_1 z_1$ and of $a_2 z_2$. The characteristic function of $a_i z_i$ has logarithm equal to

$$i\langle h, a_i m_i \rangle - \frac{1}{2}\langle a_i^2 Ch, h \rangle.$$

Adding this for $i = 1, 2$ gives the logarithm of the characteristic function of z , from which its mean and covariance may be read off.

For the second result, we note that the characteristic function of $a + Lz$ is the expectation of the exponential of

$$i\langle h, a + Lz \rangle = i\langle h, a \rangle + i\langle L^T h, z \rangle.$$

Using the properties of the characteristic functions of z , we deduce that the logarithm of the characteristic function of $a + Lz$ is equal to

$$i\langle h, a \rangle + i\langle L^T h, m \rangle - \frac{1}{2}\langle CL^T h, L^T h \rangle.$$

This may be rewritten as

$$i\langle h, a + Lm \rangle - \frac{1}{2}\langle LCL^T h, h \rangle,$$

which is the logarithm of the characteristic function of $N(a + Lm, LCL^T)$, as required. \square

We finish by stating a lemma whose proof is straightforward, given the foregoing material in this section, and left as an exercise.

Lemma 1.6. *Define*

$$I(v) := \frac{1}{2}\langle (v - m), L(v - m) \rangle$$

with $L \in \mathbb{R}_{\text{sym}}^{\ell \times \ell}$ satisfying $L > 0$ and $m \in \mathbb{R}^\ell$. Then $\exp(-I(v))$ can be normalized to produce the pdf of the Gaussian random variable $N(m, L^{-1})$ on \mathbb{R}^ℓ . The matrix L is known as the precision matrix of the Gaussian random variable.

1.1.3. Conditional and Marginal Distributions

Let $(a, b) \in \mathbb{R}^\ell \times \mathbb{R}^m$ denote a jointly varying random variable.

Definition 1.7. *The marginal pdf $\mathbb{P}(a)$ of a is given in terms of the pdf $\mathbb{P}(a, b)$ of (a, b) by*

$$\mathbb{P}(a) = \int_{\mathbb{R}^m} \mathbb{P}(a, b) db.$$



Remark 1.8. *With this definition, for $A \subset \mathcal{B}(\mathbb{R}^\ell)$,*

$$\begin{aligned} \mathbb{P}(a \in A) &= \mathbb{P}\left((a, b) \in A \times \mathbb{R}^m\right) = \int_A \int_{\mathbb{R}^m} \mathbb{P}(a, b) da db \\ &= \int_A \left(\int_{\mathbb{R}^m} \mathbb{P}(a, b) db \right) da = \int_A \mathbb{P}(a) da. \end{aligned}$$

Thus the marginal pdf $\mathbb{P}(a)$ is indeed the pdf for a in situations in which we have no information about the random variable b other than that it is in \mathbb{R}^m . \spadesuit

We now consider the situation that is the extreme opposite of the marginal situation. To be precise, we assume that we know *everything* about the random variable b : we have observed it and know what value it takes. This leads to consideration of the random variable a given that we know the value taken by b ; we write $a|b$ for a given b . The following definition is then natural:

Definition 1.9. *The conditional pdf $\mathbb{P}(a|b)$ of $a|b$ is defined by*

$$\mathbb{P}(a|b) = \frac{\mathbb{P}(a, b)}{\mathbb{P}(b)}. \tag{1.5}$$



Remark 1.10. Conditioning a jointly varying random variable can be useful in computing probabilities, as the following calculation demonstrates:

$$\begin{aligned}\mathbb{P}\left((a, b) \in A \times B\right) &= \int_A \int_B \mathbb{P}(a, b) da db \\ &= \int_A \int_B \mathbb{P}(a|b)\mathbb{P}(b) da db \\ &= \int_B \underbrace{\left(\int_A \mathbb{P}(a|b) da\right)}_{=:I_1} \underbrace{\mathbb{P}(b) db}_{=:I_2}.\end{aligned}$$

Given b , I_1 computes the probability that a is in A . Then I_2 denotes averaging over given outcomes of b in B .



1.1.4. Bayes's Formula

By Definition 1.9, we have

$$\mathbb{P}(a, b) = \mathbb{P}(a|b)\mathbb{P}(b), \quad (1.6a)$$

$$\mathbb{P}(a, b) = \mathbb{P}(b|a)\mathbb{P}(a). \quad (1.6b)$$

Equating and rearranging, we obtain **Bayes's formula**, which states that

$$\mathbb{P}(a|b) = \frac{1}{\mathbb{P}(b)} \mathbb{P}(b|a)\mathbb{P}(a). \quad (1.7)$$

The importance of this formula is apparent in situations in which $\mathbb{P}(a)$ and $\mathbb{P}(b|a)$ are individually easy to write down. Then $\mathbb{P}(a|b)$ may be identified easily, too.

Example 1.11. Let $(a, b) \in \mathbb{R} \times \mathbb{R}$ be a jointly varying random variable specified via

$$\begin{aligned}a &\sim N(m, \sigma^2), \quad \mathbb{P}(a); \\ b|a &\sim N(f(a), \gamma^2), \quad \mathbb{P}(b|a).\end{aligned}$$

Notice that by equation (1.5), $\mathbb{P}(a, b)$ is defined via two Gaussian distributions. In fact, we have

$$\mathbb{P}(a, b) = \frac{1}{2\pi\gamma\sigma} \exp\left(-\frac{1}{2\gamma^2}|b - f(a)|^2 - \frac{1}{2\sigma^2}|a - m|^2\right).$$

Unless $f(\cdot)$ is linear, this is not the pdf of a Gaussian distribution. Integrating over a , we obtain, from the definition of the marginal pdf of b ,

$$\mathbb{P}(b) = \frac{1}{2\pi\gamma\sigma} \int_{\mathbb{R}} \exp\left(-\frac{1}{2\gamma^2}|b - f(a)|^2 - \frac{1}{2\sigma^2}|a - m|^2\right) da.$$

Using equation (1.6) then shows that

$$\mathbb{P}(a|b) = \frac{1}{\mathbb{P}(b)} \times \frac{1}{2\pi\pi\gamma\sigma} \exp\left(-\frac{1}{2\gamma^2}|b - f(a)|^2 - \frac{1}{2\sigma^2}|a - m|^2\right).$$

Note that $a|b$, like (a, b) , is not Gaussian. Thus for both (a, b) and $a|b$, we have constructed a non-Gaussian pdf in a simple fashion from the knowledge of the two Gaussians a and $b|a$. ♠

When Bayes's formula (1.7) is used in statistics, then b is typically observed data, and a is the unknown about which we wish to find information using the data. In this context, we refer to $\mathbb{P}(a)$ as the **prior**, to $\mathbb{P}(b|a)$ as the **likelihood**, and to $\mathbb{P}(a|b)$ as the **posterior**. The beauty of Bayes's formula as a tool in applied mathematics is that the likelihood is often easy to determine explicitly, given reasonable assumptions on the observational noise, while there is considerable flexibility inherent in modeling prior knowledge via probabilities to give the prior. Combining the prior and likelihood as in (1.7) gives the posterior, which is the random variable of interest; while the probability distributions used to define the likelihood $\mathbb{P}(b|a)$ (via a probability density on the data space) and prior $\mathbb{P}(a)$ (via a probability on the space of unknowns) may be quite simple, the resulting posterior probability distribution can be very complicated. A second key point to note about Bayes's formula in this context is that $\mathbb{P}(b)$, which normalizes the posterior to a pdf, may be hard to determine explicitly, but algorithms exist to find information from the posterior without knowing this normalization constant. We return to this point in subsequent chapters.

1.1.5. Independence

Consider the jointly varying random variable $(a, b) \in \mathbb{R}^\ell \times \mathbb{R}^m$. The random variables a and b are said to be *independent* if

$$\mathbb{P}(a, b) = \mathbb{P}(a)\mathbb{P}(b).$$

In this case, for $f : \mathbb{R}^\ell \rightarrow \mathbb{R}^{\ell'}$ and $g : \mathbb{R}^m \rightarrow \mathbb{R}^{m'}$,

$$\mathbb{E}f(a)g(b)^T = (\mathbb{E}f(a)) \times (\mathbb{E}g(b)^T),$$

since

$$\mathbb{E}f(a)g(b)^T = \int_{\mathbb{R}^\ell \times \mathbb{R}^m} f(a)g(b)^T \mathbb{P}(a)\mathbb{P}(b) da db = \left(\int_{\mathbb{R}^\ell} f(a)\mathbb{P}(a) da \right) \left(\int_{\mathbb{R}^m} g(b)^T \mathbb{P}(b) db \right).$$

An i.i.d. (independent, identically distributed) sequence $\{\xi_j\}_{j \in \mathbb{N}}$ is one that satisfies the following conditions.³

- Each ξ_j is distributed according to the same pdf ρ .
- ξ_j is independent of ξ_k for $j \neq k$.

If \mathbb{J} is a subset of \mathbb{N} with finite cardinality, then this i.i.d. sequence satisfies

$$\mathbb{P}(\{\xi_j\}_{j \in \mathbb{J}}) = \prod_{j \in \mathbb{J}} \rho(\xi_j).$$

³ This discussion is easily generalized to $j \in \mathbb{Z}^+$.

1.2 Dynamical Systems

We will discuss data assimilation in the context of both discrete-time and continuous-time dynamical systems. In this section, we introduce some basic facts about such dynamical systems.

1.2.1. Iterated Maps

Let $\in C(\mathbb{R}^\ell, \mathbb{R}^\ell)$. We will frequently be interested in the iterated map, or discrete-time dynamical system, defined by

$$v_{j+1} = \Psi(v_j), \quad v_0 = u,$$

and in studying properties of the sequence $\{v_j\}_{j \in \mathbb{Z}^+}$. A *fixed point* of the map is a point v_∞ that satisfies $v_\infty = \Psi(v_\infty)$; initializing the map at $u = v_\infty$ will result in a sequence satisfying $v_j = v_\infty$ for all $j \in \mathbb{Z}^+$.

Example 1.12. Let

$$\Psi(v) = \lambda v + a.$$

Then

$$v_{j+1} = \lambda v_j + a, \quad v_0 = u.$$

By induction, we see that for $\lambda \neq 1$,

$$v_j = \lambda^j u + a \sum_{i=0}^{j-1} \lambda^i = \lambda^j u + a \frac{1 - \lambda^j}{1 - \lambda}.$$

Thus if $|\lambda| < 1$, then

$$v_j \rightarrow \frac{a}{1 - \lambda} \quad \text{as } j \rightarrow \infty.$$

The limiting value $\frac{a}{1 - \lambda}$ is a fixed point of the map. ♠

Remark 1.13. In the preceding example, the long-term dynamics of the map, for $|\lambda| < 1$, is described by convergence to a fixed point. Far more complex behavior is, of course, possible; we will explore such complex behavior in the next chapter. ♠

The following result is known as the (discrete-time) Gronwall lemma.

Lemma 1.14. Let $\{v_j\}_{j \in \mathbb{Z}^+}$ be a positive sequence and (λ, a) a pair of real numbers with $\lambda > 0$. Then if

$$v_{j+1} \leq \lambda v_j + a, \quad j = 0, 1, \dots,$$

it follows that

$$v_j \leq \lambda^j v_0 + a \frac{1 - \lambda^j}{1 - \lambda}, \quad \lambda \neq 1,$$

and

$$v_j \leq v_0 + ja, \quad \lambda = 1.$$

Proof We prove the case $\lambda \neq 1$; the case $\lambda = 1$ may be proved similarly. We proceed by induction. The result clearly holds for $j = 0$. Assume that the result is true for $j = J$. Then

$$\begin{aligned} v_{J+1} &\leq \lambda v_J + a \\ &\leq \lambda \left(\lambda^J v_0 + a \frac{1 - \lambda^J}{1 - \lambda} \right) + a \\ &= \lambda^{J+1} v_0 + a \frac{\lambda - \lambda^{J+1}}{1 - \lambda} + a \frac{1 - \lambda}{1 - \lambda} \\ &= \lambda^{J+1} v_0 + a \frac{1 - \lambda^{J+1}}{1 - \lambda}. \end{aligned}$$

This establishes the inductive step, and the proof is complete. \square

We will also be interested in stochastic dynamical systems of the form

$$v_{j+1} = \Psi(v_j) + \xi_j, \quad v_0 = u,$$

where $\xi = \{\xi_j\}_{j \in \mathbb{N}}$ is an i.i.d. sequence of random variables on \mathbb{R}^ℓ , and u is a random variable on \mathbb{R}^ℓ , independent of ξ .

Example 1.15. This is a simple but important one-dimensional (i.e., $\ell = 1$) example. Let $|\lambda| < 1$, and let

$$\begin{aligned} v_{j+1} &= \lambda v_j + \xi_j, \quad \xi_j \sim N(0, \sigma^2) \text{ i.i.d.}, \\ v_0 &\sim N(m_0, \sigma_0^2). \end{aligned}$$

By induction,

$$v_j = \lambda^j v_0 + \sum_{i=0}^{j-1} \lambda^{j-i-1} \xi_i.$$

Thus v_j is Gaussian, as a linear transformation of Gaussians; see Lemma 1.5. Furthermore, using independence of the initial condition from the sequence ξ , we obtain

$$\begin{aligned} m_j &:= \mathbb{E}v_j = \lambda^j m_0, \\ \sigma_j^2 &:= \mathbb{E}(v_j - m_j)^2 = \lambda^{2j} \mathbb{E}(v_0 - m_0)^2 + \sum_{i=0}^{j-1} \lambda^{2j-2i-2} \sigma^2 \\ &= \lambda^{2j} \sigma_0^2 + \sigma^2 \sum_{i=0}^{j-1} \lambda^{2i} = \lambda^{2j} \sigma_0^2 + \sigma^2 \frac{1 - \lambda^{2j}}{1 - \lambda^2}. \end{aligned}$$

Since $|\lambda| < 1$, we deduce that $m_j \rightarrow 0$ and $\sigma_j^2 \rightarrow \sigma^2(1 - \lambda^2)^{-1}$. Thus the sequence of Gaussians generated by this stochastic dynamical system has a limit, which is a centered Gaussian with variance larger than the variance of ξ_1 , unless $\lambda = 0$. \spadesuit

1.2.2. Differential Equations

Let $f \in C^1(\mathbb{R}^\ell, \mathbb{R}^\ell)$ and consider the ordinary differential equation (ODE)

$$\frac{dv}{dt} = f(v), \quad v(0) = u.$$

Assume that a solution exists for all $u \in \mathbb{R}^\ell$, $t \in \mathbb{R}^+$; for a given u , this solution is then an element of the space $C^1(\mathbb{R}^+; \mathbb{R}^\ell)$. In this situation, the ODE generates a continuous-time dynamical system. We are interested in properties of the function v . An *equilibrium point* $v_\infty \in \mathbb{R}^\ell$ is a point for which $f(v_\infty) = 0$. Initializing the equation at $u = v_\infty$ results in a solution $v(t) = v_\infty$ for all $t \geq 0$.

Example 1.16. Let $f(v) = -\alpha v + \beta$. Then

$$e^{\alpha t} \left(\frac{dv}{dt} + \alpha v \right) = \beta e^{\alpha t},$$

and so

$$\frac{d}{dt} \left(e^{\alpha t} v \right) = \frac{d}{dt} \left(\frac{\beta}{\alpha} e^{\alpha t} \right).$$

Thus

$$e^{\alpha t} v(t) - u = \frac{\beta}{\alpha} (e^{\alpha t} - 1),$$

so that

$$v(t) = e^{-\alpha t} u + \frac{\beta}{\alpha} (1 - e^{-\alpha t}).$$

If $\alpha > 0$, then

$$v(t) \rightarrow \frac{\beta}{\alpha} \quad \text{as } t \rightarrow \infty.$$

Note that $v_\infty := \frac{\beta}{\alpha}$ is the unique equilibrium point of the equation. ♠

Remark 1.17. In the preceding example, the long-term dynamics of the ODE, for $\alpha > 0$, is described by convergence to an equilibrium point. As in discrete time, far more complex behavior is, of course, possible; we will explore this possibility in the next chapter. ♠

If the differential equation has a solution for every $u \in \mathbb{R}^\ell$ and every $t \in \mathbb{R}^+$, then there is a one-parameter semigroup of operators $\Psi(\cdot; t)$, parameterized by time $t \geq 0$, with the properties that

$$v(t) = \Psi(u; t), \quad t \in (0, \infty), \tag{1.10a}$$

$$\Psi(u; t+s) = \Psi(\Psi(u; s); t), \quad t, s \in \mathbb{R}^+, u \in \mathbb{R}^\ell, \tag{1.10b}$$

$$\Psi(u; 0) = u \in \mathbb{R}^\ell. \tag{1.10c}$$

We call $\Psi(\cdot; \cdot)$ the solution operator for the ODE. In this scenario, we can consider the iterated map defined by $\Psi(\cdot) = \Psi(\cdot; h)$, for some fixed $h > 0$, thereby linking the discrete-time iterated maps with continuous-time ODEs.

Example 1.18. (Example 1.16, continued) Let

$$\Psi(u; t) = e^{-\alpha t} u + \frac{\beta}{\alpha} (1 - e^{-\alpha t}),$$

which is the solution operator for the equation in that $v(t) = \Psi(u; t)$. Clearly, $\Psi(u; 0) = u$. Also

$$\begin{aligned}
\Psi(u; t+s) &= e^{-\alpha t} e^{-\alpha s} u + \frac{\beta}{\alpha} (1 - e^{-\alpha t} e^{-\alpha s}) \\
&= e^{-\alpha t} \left(e^{-\alpha s} u + \frac{\beta}{\alpha} (1 - e^{-\alpha s}) \right) + \frac{\beta}{\alpha} (1 - e^{-\alpha t}) \\
&= \Psi(\Psi(u; s); t).
\end{aligned}$$



The following result is known as the (continuous-time) Gronwall lemma.

Lemma 1.19. *Let $z \in C^1(\mathbb{R}^+, \mathbb{R})$ satisfy*

$$\frac{dz}{dt} \leq az + b, \quad z(0) = z_0,$$

for some $a, b \in \mathbb{R}$. Then

$$z(t) \leq e^{at} z_0 + \frac{b}{a} (e^{at} - 1).$$

Proof Multiplying both sides of the given identity by e^{-at} , we obtain

$$e^{-at} \left(\frac{dz}{dt} - az \right) \leq be^{-at},$$

which implies that

$$\frac{d}{dt} (e^{-at} z) \leq be^{-at}.$$

Therefore,

$$e^{-at} z(t) - z(0) \leq \frac{b}{a} (1 - e^{-at}),$$

so that

$$z(t) \leq e^{at} z_0 + \frac{b}{a} (e^{at} - 1).$$



1.2.3. Long-Time Behavior

We consider the long-time behavior of discrete-time dynamical systems. The ideas are easily generalized to continuous-time dynamical systems—ODEs—and indeed, our example will demonstrate such a generalization. To facilitate our definitions, we now extend Ψ to act on Borel subsets of \mathbb{R}^ℓ . Note that currently, $\Psi : \mathbb{R}^\ell \rightarrow \mathbb{R}^\ell$; we extend to $\Psi : \mathcal{B}(\mathbb{R}^\ell) \rightarrow \mathcal{B}(\mathbb{R}^\ell)$ via

$$\Psi(A) = \bigcup_{u \in A} \Psi(u), \quad A \in \mathcal{B}(\mathbb{R}^\ell).$$

For both $\Psi : \mathbb{R}^\ell \rightarrow \mathbb{R}^\ell$ and $\Psi : \mathcal{B}(\mathbb{R}^\ell) \rightarrow \mathcal{B}(\mathbb{R}^\ell)$, we denote by

$$\Psi^{(j)} = \Psi \circ \dots \circ \Psi$$

the j -fold composition of Ψ with itself. In the following, let $B(0, R)$ denote the ball of radius R in \mathbb{R}^ℓ , in the Euclidean norm, centered at the origin.

Definition 1.20. A discrete-time dynamical system has a bounded absorbing set $\mathcal{B}_{\text{abs}} \subset \mathbb{R}^\ell$ if for every $R > 0$, there exists $J = J(R)$ such that

$$\Psi^{(J)}(B(0, R)) \subset \mathcal{B}_{\text{abs}}, \quad \forall j \geq J.$$



Remark 1.21. The definition of absorbing set is readily generalized to continuous-time dynamical systems; this is left as an exercise for the reader.



Example 1.22. Consider an ODE for which there exist $\alpha, \beta > 0$ such that

$$\langle f(v), v \rangle \leq \alpha - \beta|v|^2, \quad \forall v \in \mathbb{R}^\ell.$$

Then

$$\frac{1}{2} \frac{d}{dt}|v|^2 = \left\langle v, \frac{dv}{dt} \right\rangle = \langle v, f(v) \rangle \leq \alpha - \beta|v|^2.$$

Applying the Gronwall lemma (Lemma 1.19) gives

$$|v(t)|^2 \leq e^{-2\beta t} |v(0)|^2 + \frac{\alpha}{\beta} (1 - e^{-2\beta t}).$$

Hence if $|v(0)|^2 \leq R$, then

$$|v(t)|^2 \leq 2 \frac{\alpha}{\beta} \quad \forall t \geq T : e^{-2\beta t} R^2 \leq \frac{\alpha}{\beta}.$$

Therefore, the set $\mathcal{B}_{\text{abs}} = B\left(0, \sqrt{\frac{2\alpha}{R}}\right)$ is absorbing for the ODE (with the generalization of the above definition of absorbing set to continuous time, as in Remark 1.21).

If $v_j = v(jh)$, so that $\Psi(\cdot) = \Psi(\cdot; h)$ and $v_{j+1} = \Psi(v_j)$, then

$$|v_j|^2 \leq 2 \frac{\alpha}{\beta} \quad \forall J \geq \frac{T}{h},$$

where T is as in the ODE case. Hence $\mathcal{B}_{\text{abs}} = B\left(0, \sqrt{\frac{2\alpha}{R}}\right)$ is also an absorbing set for the iterated map associated with the ODE.



Definition 1.23. When the discrete-time dynamical system has a bounded absorbing set, \mathcal{B}_{abs} we define the global attractor \mathcal{A} to be

$$\mathcal{A} = \overline{\bigcup_{k \geq 0} \overline{\bigcup_{j \geq k} \Psi^{(j)}(\mathcal{B}_{\text{abs}})}}.$$



This object captures all the long-time dynamics of the dynamical system. As for the absorbing set itself, this definition is readily generalized to continuous time.

1.2.4. Controlled Dynamical Systems

It is frequently of interest to add a *controller* $w = \{w_j\}_{j=0}^\infty$ to the discrete-time dynamical system to obtain

$$v_{j+1} = \Psi(v_j) + w_j.$$

The aim of the controller is to “steer” the dynamical system to achieve some objective. Interesting examples include the following:

- Given a point $v^* \in \mathbb{R}^\ell$ and time $J \in \mathbb{Z}^+$, choose w such that $v_J = v^*$.
- Given an open set B and time $J \in \mathbb{Z}^+$, choose w such that $v_j \in B$ for all $j \geq J$.
- Given $y = \{y_j\}_{j \in \mathbb{N}}$, where $y_j \in \mathbb{R}^m$, and given a function $h : \mathbb{R}^\ell \rightarrow \mathbb{R}^m$, choose w to keep $|y_j - h(v_j)|$ small in some sense.

The third option is most relevant in the context of data assimilation, and so we focus on it. In this context, we will consider controllers of the form $w_j = K(y_j - h(v_j))$, so that

$$v_{j+1} = \Psi(v_j) + K(y_j - h(v_j)). \quad (1.11)$$

A key question is then how to choose K to ensure the desired property. We present a simple example that illustrates this.

Example 1.24. Let $\ell = m = 1$, $\Psi(v) = \lambda v$ and $h(v) = v$. We assume that the data $\{y_j\}_{j \in \mathbb{N}}$ is given by $y_{j+1} = v_{j+1}^\dagger$, where $v_{j+1}^\dagger = \lambda v_j^\dagger$. Thus the data is itself generated by the uncontrolled dynamical system. We wish to use the controller to ensure that the solution of the controlled system is close to the data $\{y_j\}_{j \in \mathbb{N}}$ generated by the uncontrolled dynamical system, and hence to the solution of the uncontrolled dynamical system itself.

Consider the controlled dynamical system

$$\begin{aligned} v_{j+1} &= \Psi(v_j) + K(y_j - h(v_j)) \\ &= \lambda v_j + \underbrace{K(y_j - v_j)}_{w_j}, \quad j \geq 1, \end{aligned}$$

and assume that $v_0 \neq v_0^\dagger$. We are interested in whether v_j approaches v_j^\dagger as $j \rightarrow \infty$.

To this end, suppose that K is chosen such that $|\lambda - K| < 1$. Then note that

$$v_{j+1}^\dagger = \lambda v_j^\dagger + K \underbrace{(y_j - v_j^\dagger)}_{=0}.$$

Hence $e_j = v_j - v_j^\dagger$ satisfies

$$e_{j+1} = (\lambda - K)e_j$$

and

$$|e_{j+1}| = |\lambda - K||e_j|.$$

Since we have chosen K such that $|\lambda - K| < 1$, we have $|e_j| \rightarrow 0$ as $j \rightarrow \infty$. Thus the controlled dynamical system approaches the solution of the uncontrolled dynamical system as $j \rightarrow \infty$. This is prototypical of certain data-assimilation algorithms that we will study in Chapter 4. ♠

It is also of interest to consider continuous-time controllers $\{w(t)\}_{t \geq 0}$ for differential equations

$$\frac{dv}{dt} = f(v) + w.$$

Again, the goal is to choose w to achieve some objective analogous to those described in discrete time.

1.3 Probability Metrics

Since we will frame data assimilation in terms of probability, natural measures of robustness of the problem will require the idea of distance between probability measures. Here we introduce basic metric properties, and then some specific distances on probability measures and their properties.

1.3.1. Metric Properties

Definition 1.25. A metric on a set X is a function $d : X \times X \rightarrow \mathbb{R}^+$ (distance) satisfying the following properties:

- coincidence: $d(x, y) = 0$ iff $x = y$;
- symmetry: $d(x, y) = d(y, x)$;
- triangle: $d(x, z) \leq d(x, y) + d(y, z)$.



Example 1.26. Let $X = \mathbb{R}^\ell$, viewed as a normed vector space with norm $\|\cdot\|$; for example, we might take $\|\cdot\| = |\cdot|$, the Euclidean norm. Then the function $d : \mathbb{R}^\ell \times \mathbb{R}^\ell \rightarrow \mathbb{R}^+$ given by $d(x, y) := \|x - y\|$ defines a metric. Indeed, from the properties of norms:

- $\|x - y\| = 0$ iff $x = y$;
- $\|x - y\| = \|y - x\|$;
- $\|x - z\| = \|x - y + y - z\| \leq \|x - y\| + \|y - z\|$.



1.3.2. Metrics on Spaces of Probability Measures

Let \mathcal{M} denote the space of probability measures on \mathbb{R}^ℓ with strictly positive Lebesgue density on \mathbb{R}^ℓ . Throughout this section, we let μ and μ' be two probability measures on \mathcal{M} , and let ρ and ρ' denote the corresponding densities; recall that we assume that these densities are positive everywhere, in order to simplify the presentation. We define two useful metrics on probability measures.

Definition 1.27. *The total-variation distance on \mathcal{M} is defined by*

$$\begin{aligned} d_{\text{TV}}(\mu, \mu') &= \frac{1}{2} \int_{\mathbb{R}^\ell} |\rho(u) - \rho'(u)| du \\ &= \frac{1}{2} \mathbb{E}^\mu \left| 1 - \frac{\rho'(u)}{\rho(u)} \right|. \end{aligned}$$



Thus the total-variation distance is half of the L^1 norm of the difference of the two pdfs. Note that clearly, $d_{\text{TV}}(\mu, \mu') \geq 0$. Also,

$$\begin{aligned} d_{\text{TV}}(\mu, \mu') &\leq \frac{1}{2} \int_{\mathbb{R}^\ell} |\rho(u)| du + \frac{1}{2} \int_{\mathbb{R}^\ell} |\rho'(u)| du \\ &= \frac{1}{2} \int_{\mathbb{R}^\ell} \rho(u) du + \frac{1}{2} \int_{\mathbb{R}^\ell} \rho'(u) du \\ &= 1. \end{aligned}$$

Note also that d_{TV} may be characterized as

$$d_{\text{TV}}(\mu, \mu') = \frac{1}{2} \sup_{|f|_\infty \leq 1} |\mathbb{E}^\mu(f) - \mathbb{E}^{\mu'}(f)| = \frac{1}{2} \sup_{|f|_\infty \leq 1} |\mu(f) - \mu'(f)|, \quad (1.12)$$

where we have used the convention that $\mu(f) = \mathbb{E}^\mu(f) = \int_{\mathbb{R}^\ell} f(v) \mu(dv)$ and $|f|_\infty = \sup_u |f(u)|$.

Definition 1.28. *The Hellinger distance on \mathcal{M} is defined by*

$$\begin{aligned} d_{\text{Hell}}(\mu, \mu') &= \left(\frac{1}{2} \int_{\mathbb{R}^\ell} \left(\sqrt{\rho(u)} - \sqrt{\rho'(u)} \right)^2 du \right)^{1/2} \\ &= \left(\frac{1}{2} \mathbb{E}^\mu \left(1 - \sqrt{\frac{\rho'(u)}{\rho(u)}} \right)^2 \right)^{1/2}. \end{aligned}$$



Thus the Hellinger distance is a multiple of the L^2 distance between the square roots of the two pdfs. Again, clearly $d_{\text{Hell}}(\mu, \mu') \geq 0$. Also,

$$d_{\text{Hell}}(\mu, \mu')^2 \leq \frac{1}{2} \int_{\mathbb{R}^\ell} (\rho(u) + \rho'(u)) du = 1.$$

We also note that the Hellinger and total-variation distances can be written in a symmetric way and that they satisfy the triangle inequality—they are indeed valid distance metrics on the space of probability measures.

Lemma 1.29. *The total variation and Hellinger distances satisfy*

$$0 \leq \frac{1}{\sqrt{2}} d_{\text{TV}}(\mu, \mu') \leq d_{\text{Hell}}(\mu, \mu') \leq d_{\text{TV}}(\mu, \mu')^{1/2} \leq 1.$$

Proof The upper and lower bounds of, respectively, 0 and 1 are proved above. We show first that $\frac{1}{\sqrt{2}}d_{\text{TV}}(\mu, \mu') \leq d_{\text{Hell}}(\mu, \mu')$. Indeed, by the Cauchy–Schwarz inequality,

$$\begin{aligned} d_{\text{TV}}(\mu, \mu') &= \frac{1}{2} \int_{\mathbb{R}^\ell} \left| 1 - \sqrt{\frac{\rho'(u)}{\rho(u)}} \right| \left| 1 + \sqrt{\frac{\rho'(u)}{\rho(u)}} \right| \rho(u) du \\ &\leq \left(\frac{1}{2} \int_{\mathbb{R}^\ell} \left| 1 - \sqrt{\frac{\rho'(u)}{\rho(u)}} \right|^2 \rho(u) du \right)^{1/2} \left(\frac{1}{2} \int_{\mathbb{R}^\ell} \left| 1 + \sqrt{\frac{\rho'(u)}{\rho(u)}} \right|^2 \rho(u) du \right)^{1/2} \\ &\leq d_{\text{Hell}}(\mu, \mu') \left(\int_{\mathbb{R}^\ell} \left| 1 + \frac{\rho'(u)}{\rho(u)} \right| \rho(u) du \right)^{1/2} \\ &= \sqrt{2}d_{\text{Hell}}(\mu, \mu'). \end{aligned}$$

Finally, for the inequality $d_{\text{Hell}}(\mu, \mu') \leq d_{\text{TV}}(\mu, \mu')^{1/2}$ note that

$$|\sqrt{a} - \sqrt{b}| \leq \sqrt{a} + \sqrt{b} \quad \forall a, b > 0.$$

Therefore,

$$\begin{aligned} d_{\text{Hell}}(\mu, \mu')^2 &= \frac{1}{2} \int_{\mathbb{R}^\ell} \left| 1 - \sqrt{\frac{\rho'(u)}{\rho(u)}} \right| \left| 1 - \sqrt{\frac{\rho'(u)}{\rho(u)}} \right| \rho(u) du \\ &\leq \frac{1}{2} \int_{\mathbb{R}^\ell} \left| 1 - \sqrt{\frac{\rho'(u)}{\rho(u)}} \right| \left| 1 + \sqrt{\frac{\rho'(u)}{\rho(u)}} \right| \rho(u) du \\ &= \frac{1}{2} \int_{\mathbb{R}^\ell} \left| 1 - \frac{\rho'(u)}{\rho(u)} \right| \rho(u) du \\ &= d_{\text{TV}}(\mu, \mu'). \end{aligned}$$

□

Why do we bother to introduce the Hellinger distance, rather than working with the more familiar total variation? The answer stems from the following two lemmas.

Lemma 1.30. *Let $f : \mathbb{R}^\ell \rightarrow \mathbb{R}^p$ be such that*

$$(\mathbb{E}^\mu |f(u)|^2 + \mathbb{E}^{\mu'} |f(u)|^2) < \infty.$$

Then

$$|\mathbb{E}^\mu f(u) - \mathbb{E}^{\mu'} f(u)| \leq 2(\mathbb{E}^\mu |f(u)|^2 + \mathbb{E}^{\mu'} |f(u)|^2)^{\frac{1}{2}} d_{\text{Hell}}(\mu, \mu'). \quad (1.13)$$

As a consequence,

$$|\mathbb{E}^\mu f(u) - \mathbb{E}^{\mu'} f(u)| \leq 2(\mathbb{E}^\mu |f(u)|^2 + \mathbb{E}^{\mu'} |f(u)|^2)^{\frac{1}{2}} d_{\text{TV}}(\mu, \mu')^{\frac{1}{2}}. \quad (1.14)$$

Proof In the following, all integrals are over \mathbb{R}^ℓ . Now,

$$\begin{aligned}
|\mathbb{E}^\mu f(u) - \mathbb{E}^{\mu'} f(u)| &\leq \int |f(u)| |\rho(u) - \rho'(u)| du \\
&= \int \sqrt{2} |f(u)| |\sqrt{\rho(u)} + \sqrt{\rho'(u)}| \cdot \frac{1}{\sqrt{2}} |\sqrt{\rho(u)} - \sqrt{\rho'(u)}| du \\
&\leq \left(\int 2 |f(u)|^2 |\sqrt{\rho(u)} + \sqrt{\rho'(u)}|^2 du \right)^{\frac{1}{2}} \left(\frac{1}{2} \int |\sqrt{\rho(u)} - \sqrt{\rho'(u)}|^2 du \right)^{\frac{1}{2}} \\
&\leq \left(\int 4 |f(u)|^2 (\rho(u) + \rho'(u)) du \right)^{\frac{1}{2}} \left(\frac{1}{2} \int \left(1 - \frac{\sqrt{\rho'(u)}}{\sqrt{\rho(u)}} \right)^2 \rho(u) du \right)^{\frac{1}{2}} \\
&= 2(\mathbb{E}^\mu |f(u)|^2 + \mathbb{E}^{\mu'} |f(u)|^2)^{\frac{1}{2}} d_{\text{Hell}}(\mu, \mu').
\end{aligned}$$

Thus (1.13) follows. The bound (1.14) follows from Lemma 1.29. \square

Remark 1.31. *The preceding lemma shows that if two measures μ and μ' are $\mathcal{O}(\epsilon)$ -close in the Hellinger metric, and if the function $f(u)$ is square-integrable with respect to u distributed according to μ and μ' , then expectations of $f(u)$ with respect to μ and μ' are also $\mathcal{O}(\epsilon)$ -close. It also shows that under the same assumptions on f , if two measures μ and μ' are $\mathcal{O}(\epsilon)$ -close in the total-variation metric, then expectations of $f(u)$ with respect to μ and μ' are only $\mathcal{O}(\epsilon^{\frac{1}{2}})$ -close. This second result is sharp, and to get $\mathcal{O}(\epsilon)$ -closeness of expectations using $\mathcal{O}(\epsilon)$ -closeness in the total-variation metric requires a stronger assumption on f , as we now show.* ♠

Lemma 1.32. *Assume that $|f|$ is finite almost surely with respect to both μ and μ' and denote the almost sure upper bound on $|f|$ by f_{\max} . Then*

$$|\mathbb{E}^\mu f(u) - \mathbb{E}^{\mu'} f(u)| \leq 2f_{\max} d_{\text{TV}}(\mu, \mu').$$

Proof Under the given assumption on f ,

$$\begin{aligned}
|\mathbb{E}^\mu f(u) - \mathbb{E}^{\mu'} f(u)| &\leq \int |f(u)| |\rho(u) - \rho'(u)| du \\
&\leq 2f_{\max} \left(\frac{1}{2} \int |\rho(u) - \rho'(u)| du \right) \\
&\leq 2f_{\max} \left(\frac{1}{2} \int \left| 1 - \frac{\rho'(u)}{\rho(u)} \right| \rho(u) du \right) \\
&= 2f_{\max} d_{\text{TV}}(\mu, \mu').
\end{aligned}$$

\square

The implication of the preceding two lemmas and Remark 1.31 is that it is natural to work with the Hellinger metric, rather than the total-variation metric, in considering the effect of perturbations of the measure on expectations of functions that are square-integrable but not bounded.

1.4 Probabilistic View of Dynamical Systems

Here we look at the natural connection between dynamical systems and the underlying dynamical system that they generate on probability measures. The key idea here is that the Markovian propagation of probability measures is linear, even when the underlying dynamical

system is nonlinear. This advantage of linearity is partially offset by the fact that the underlying dynamics on probability distributions is infinite-dimensional, but it is nonetheless a powerful perspective on dynamical systems. Example 1.15 provides a nice introductory example demonstrating the probability distributions carried by a stochastic dynamical system; in that case, the probability distributions are Gaussian, and we explicitly characterize their evolution through the mean and covariance. The idea of mapping probability measures under the dynamical system can be generalized, but the situation is typically more complicated, because the probability distributions are typically not Gaussian and not characterized by a finite number of parameters.

1.4.1. Markov Kernel

Definition 1.33. *The function $p : \mathbb{R}^\ell \times \mathcal{B}(\mathbb{R}^\ell) \rightarrow \mathbb{R}^+$ is a Markov kernel if the following conditions are satisfied:*

- For each $x \in \mathbb{R}^\ell$, $p(x, \cdot)$ is a probability measure on $(\mathbb{R}^\ell, \mathcal{B}(\mathbb{R}^\ell))$;
- $x \mapsto p(x, A)$ is $\mathcal{B}(\mathbb{R}^\ell)$ -measurable for all $A \in \mathcal{B}(\mathbb{R}^\ell)$.



The first condition is the key one for the material in this book: the Markov kernel at fixed x describes the probability distribution of a new point $y \sim p(x, \cdot)$. By iterating on this, we may generate a sequence of points that constitute a sample from the distribution of the Markov chain, as described below, defined by the Markov kernel. The second measurability condition ensures an appropriate mathematical setting for the problem, but an in-depth understanding of this condition is not essential for the reader of this book. In the same way that we use \mathbb{P} to denote both the probability measure and its pdf, we sometimes use $p(x, \cdot) : \mathbb{R}^\ell \rightarrow \mathbb{R}^+$, for each fixed $x \in \mathbb{R}^\ell$, to denote the corresponding pdf of the Markov kernel from the preceding definition.

Consider the stochastic dynamical system

$$v_{j+1} = \Psi(v_j) + \xi_j,$$

where $\xi = \{\xi_j\}_{j \in \mathbb{Z}^+}$ is an i.i.d. sequence distributed according to a probability measure on \mathbb{R}^ℓ with density $\rho(\cdot)$. We assume that the initial condition v_0 is possibly random, but independent of ξ . Under these assumptions on the probabilistic structure, we say that $\{v_j\}_{j \in \mathbb{Z}^+}$ is a *Markov chain*. For this Markov chain, we have

$$\mathbb{P}(v_{j+1} | v_j) = \rho(v_{j+1} - \Psi(v_j));$$

thus

$$\mathbb{P}(v_{j+1} \in A | v_j) = \int_A \rho(v_{j+1} - \Psi(v_j)) dv.$$

In fact, we can define a Markov kernel

$$p(u, A) = \int_A \rho(v - \Psi(u)) dv,$$

with the associated pdf

$$p(u, v) = \rho(v - \Psi(u)).$$

If $v_j \sim \mu_j$ with pdf ρ_j , then

$$\begin{aligned}\mu_{j+1} &= \mathbb{P}(v_{j+1} \in A) \\ &= \int_{\mathbb{R}^\ell} \mathbb{P}(v_{j+1} \in A | v_j) \mathbb{P}(v_j) dv_j \\ &= \int_{\mathbb{R}^\ell} p(u, A) \rho_j(u) du.\end{aligned}$$

And then

$$\begin{aligned}\rho_{j+1}(v) &= \int_{\mathbb{R}^\ell} p(u, v) \rho_j(u) du \\ &= \int_{\mathbb{R}^\ell} \rho(v - \Psi(u)) \rho_j(u) du.\end{aligned}$$

Furthermore, we have a linear dynamical system for the evolution of the pdf

$$\rho_{j+1} = P \rho_j, \quad (1.15)$$

where P is the integral operator

$$(P\pi)(v) = \int_{\mathbb{R}^\ell} \rho(v - \Psi(u)) \pi(u) du.$$

Example 1.34. Let $\Psi : \mathbb{R}^\ell \rightarrow \mathbb{R}^\ell$. Assume that $\xi_1 \sim N(0, \sigma^2 I)$. Then

$$\rho_{j+1}(v) = \int_{\mathbb{R}^\ell} \frac{1}{(2\pi)^{\ell/2} \sigma^\ell} \exp\left(-\frac{1}{2\sigma^2} |v - \Psi(u)|^2\right) \rho_j(u) du.$$

As $\sigma \rightarrow \infty$, we obtain the deterministic model

$$\rho_{j+1}(v) = \int_{\mathbb{R}^\ell} \delta(v - \Psi(u)) \rho_j(u) du.$$



For each integer $n \in \mathbb{N}$, we use the notation $p^n(u, \cdot)$ to denote the Markov kernel arising from n steps of the Markov chain; thus $p^1(u, \cdot) = p(u, \cdot)$. Furthermore, $p^n(u, A) = \mathbb{P}(u^{(n)} \in A | u^{(0)} = u)$.

1.4.2. Ergodicity

In many situations, we will appeal to *ergodic theorems* to extract information from sequences $\{v_j\}_{j \in \mathbb{Z}^+}$ generated by a (possibly stochastic) dynamical system. Assume that this dynamical system is invariant with respect to the probability measure μ_∞ . Then, roughly speaking, an ergodic dynamical system is one for which, for a suitable class of test functions $\varphi : \mathbb{R}^\ell \rightarrow \mathbb{R}$, and v_0 almost surely with respect to the *invariant measure* μ_∞ , the Markov chain from the previous subsection satisfies

$$\frac{1}{J} \sum_{j=1}^J \varphi(v_j) \rightarrow \int_{\mathbb{R}^\ell} \varphi(v) \mu_\infty(dv) = \mathbb{E}^{\mu_\infty} \varphi(v). \quad (1.16)$$

We say that *the time average equals the space average*. The preceding identity encodes the idea that the histogram formed by a single trajectory $\{v_j\}$ of the Markov chain looks more and more like the pdf of the underlying invariant measure. Since the convergence is almost sure with respect to the initial condition, this implies that the statistics on where the trajectory spends time is asymptotically independent of the initial condition; this is a very powerful property.

If the Markov chain has a unique *invariant density* ρ_∞ , which is a fixed point of the linear dynamical system (1.15), then it will satisfy

$$\rho_\infty = P\rho_\infty, \quad (1.17)$$

or equivalently,

$$\rho_\infty(v) = \int_{\mathbb{R}^\ell} p(u, v)\rho_\infty(u)du. \quad (1.18)$$

In the ergodic setting, this equation will have a form of uniqueness within the class of pdfs, and furthermore, it is often possible to prove, in some norm, the convergence

$$\rho_j \rightarrow \rho_\infty \quad \text{as } j \rightarrow \infty.$$

Example 1.35. Example 1.15 generates an ergodic Markov chain $\{v_j\}_{j \in \mathbb{Z}^+}$ carrying the sequence of pdfs ρ_j . Furthermore, each ρ_j is the density of a Gaussian $N(m_j, \sigma_j^2)$. If $|\lambda| < 1$, then $m_j \rightarrow 0$ and $\sigma_j^2 \rightarrow \sigma_\infty^2$, where

$$\sigma_\infty^2 = \frac{\sigma^2}{1 - \lambda^2}.$$

Thus ρ_∞ is the density of a Gaussian $N(0, \sigma_\infty^2)$. We then have

$$\frac{1}{J} \sum_{j=1}^J \varphi(v_j) = \frac{1}{J} \sum_{j=1}^J \varphi \left(\lambda^j v_0 + \sum_{i=1}^{J-1} \lambda^{j-i-1} \xi_i \right) \rightarrow \int_{\mathbb{R}} \rho_\infty(v) \varphi(v) dv.$$



1.4.3. Bayes's Formula as a Map

Recall that Bayes's formula states that

$$\mathbb{P}(a|b) = \frac{1}{\mathbb{P}(b)} \mathbb{P}(b|a) \mathbb{P}(a).$$

This may be viewed as a map from $\mathbb{P}(a)$ (what we know about a a priori, the prior) to $\mathbb{P}(a|b)$ (what we know about a once we have observed the variable b , the posterior.) Since

$$\mathbb{P}(b) = \int_{\mathbb{R}^\ell} \mathbb{P}(b|a) \mathbb{P}(a) da,$$

we see that

$$\mathbb{P}(a|b) = \frac{\mathbb{P}(b|a) \mathbb{P}(a)}{\int_{\mathbb{R}^\ell} \mathbb{P}(b|a) \mathbb{P}(a) da} =: L\mathbb{P}(a).$$

Here L is a *nonlinear* map that takes the pdf $\mathbb{P}(a)$ into $\mathbb{P}(a|b)$. We use the letter L to highlight the fact that the map is defined, in the context of Bayesian statistics, using the likelihood to map prior to posterior.

1.5 Bibliography

- For background material on probability, as covered in Section 1.1, the reader is directed to the elementary textbook [23] and to the more advanced texts [60, 148] for further material (for example, the definition of measurable.) The book [113], together with the references therein, provides an excellent introduction to Markov chains. The book [108] is a comprehensive study of ergodicity for Markov chains; the central use of Lyapunov functions will make it particularly accessible to readers with a background in dynamical systems. Note also that Theorem 3.3 contains a basic ergodic result for Markov chains.
- Section 1.2 concerns dynamical systems and stochastic dynamical systems. The deterministic setting is discussed in numerous textbooks, such as [61, 147], with more advanced material, related to infinite-dimensional problems, covered in [137]. The ergodicity of stochastic dynamical systems is presented in [7], and targeted treatments based on the small-noise scenario include [53, 14]. The book [134] contains elementary chapters on dynamical systems, and the book chapter [73] contains related material in the context of stochastic dynamical systems. For the subject of control theory, the reader is directed to [149], which has a particularly good exposition of the linear theory, and [130] for the nonlinear setting.
- Probability metrics are the subject of Section 1.3, and the survey paper [57] provides a very readable introduction to this subject, together with references to the wider literature.
- Viewing (stochastic) dynamical systems as generating a dynamical system on the probability measure that they carry is an enormously powerful way of thinking. The reader is directed to the books [145] and [10] for overviews of this subject and further references.

1.6 Exercises

1. Consider the ODE

$$\frac{dv}{dt} = v - v^3, \quad v(0) = v_0.$$

By finding the exact solution, determine the one-parameter semigroup $\Psi(\cdot; t)$ with properties (1.10).

2. Consider a jointly varying random variable $(a, b) \in \mathbb{R}^2$ defined as follows: $a \sim N(0, \sigma^2)$ and $b|a \sim N(a, \gamma^2)$. Find a formula for the probability density function of (a, b) , using (1.5b), and demonstrate that the random variable is a Gaussian with mean and covariance that you should specify. Using (1.7), find a formula for the probability density function of $a|b$; again demonstrate that the random variable is a Gaussian with mean and covariance that you should specify.
3. Consider two Gaussian densities on \mathbb{R} : $N(m_1, \sigma_1^2)$ and $N(m_2, \sigma_2^2)$. Show that the Hellinger distance between them is given by

$$d_{\text{Hell}}(\mu, \mu')^2 = 1 - \sqrt{\exp\left(-\frac{(m_1 - m_2)^2}{2(\sigma_1^2 + \sigma_2^2)}\right) \frac{2\sigma_1\sigma_2}{(\sigma_1^2 + \sigma_2^2)}}.$$

4. Consider two Gaussian measures on \mathbb{R} : $N(m_1, \sigma_1^2)$ and $N(m_2, \sigma_2^2)$. Show that the total-variation distance between the measures tends to zero as $m_2 \rightarrow m_1$ and $\sigma_2^2 \rightarrow \sigma_1^2$.
5. The Kullback–Leibler divergence between two measures μ' and μ , with pdfs ρ' and ρ respectively, is

$$D_{\text{KL}}(\mu' \parallel \mu) = \int \log\left(\frac{\rho(x)'}{\rho(x)}\right) \rho'(x) dx.$$

Does D_{KL} define a metric on probability measures? Justify your answer. Consider two Gaussian densities on \mathbb{R} : $N(m_1, \sigma_1^2)$ and $N(m_2, \sigma_2^2)$. Show that the Kullback–Leibler divergence between them is given by

$$D_{\text{KL}}(\mu_1 \parallel \mu_2) = \ln\left(\frac{\sigma_2}{\sigma_1}\right) + \frac{1}{2}\left(\frac{\sigma_1^2}{\sigma_2^2} - 1\right) + \frac{(m_2 - m_1)^2}{2\sigma_2^2}.$$

6. Assume that two measures μ and μ' have positive Lebesgue densities ρ and ρ' respectively. Prove the bounds

$$d_{\text{Hell}}(\mu, \mu')^2 \leq \frac{1}{2} D_{\text{KL}}(\mu \parallel \mu') , \quad d_{\text{TV}}(\mu, \mu')^2 \leq D_{\text{KL}}(\mu \parallel \mu'),$$

where the Kullback–Leibler divergence D_{KL} is defined in the preceding exercise.

7. Consider the stochastic dynamical system of Example 1.15. Find explicit formulas for the maps $m_j \mapsto m_{j+1}$ and $\sigma_j^2 \mapsto \sigma_{j+1}^2$.
8. Directly compute the mean and covariance of $w = a + Lz$ if z is Gaussian $N(m, C)$, without using the characteristic function. Verify that you obtain the same result as in Lemma 1.5.
9. Prove Lemma 1.6.
10. Generalize Definitions 1.20 and 1.23 to continuous time, as suggested in Remark 1.21.

Chapter 2

Discrete Time: Formulation

In this chapter, we introduce the mathematical framework for discrete-time data assimilation. Section 2.1 describes the mathematical models we use for the underlying signal, which we wish to recover, and for the data, which we use for the recovery. In Section 2.2, we introduce a number of examples used throughout the text to illustrate the theory. Sections 2.3 and 2.4 respectively describe two key problems related to the conditioning of the signal v on the data y , namely **smoothing** and **filtering**; in Section 2.5, we describe how these two key problems are related. Section 2.6 proves that the smoothing problem is well posed and, using the connection to filtering described in Section 2.5, that the filtering problem is well posed; here well-posedness refers to continuity of the desired conditioned probability distribution with respect to the observed data. Section 2.7 discusses approaches to evaluating the quality of data-assimilation algorithms. In Section 2.8, we describe various illustrations of the foregoing theory and conclude the chapter with Section 2.9, devoted to a bibliographical overview, and Section 2.10, containing exercises.

2.1 Setup

We assume throughout this book that $\Psi \in C(\mathbb{R}^n, \mathbb{R}^n)$, and we consider the Markov chain $v = \{v_j\}_{j \in \mathbb{Z}^+}$ defined by the random map

$$v_{j+1} = \Psi(v_j) + \xi_j, \quad j \in \mathbb{Z}^+, \tag{2.1a}$$

$$v_0 \sim N(m_0, C_0), \tag{2.1b}$$

where $\xi = \{\xi_j\}_{j \in \mathbb{Z}^+}$ is an i.i.d. sequence, with $\xi_0 \sim N(0, \Sigma)$ and $\Sigma > 0$. Because (v_0, ξ) is a random variable, so too is the solution sequence $\{v_j\}_{j \in \mathbb{Z}^+}$: the **signal**, which determines the state of the system at each discrete time instance. For simplicity, we assume that v_0 and ξ are independent. The probability distribution of the random variable v quantifies the uncertainty in predictions arising from this **stochastic dynamics** model.

In many applications, models such as (2.1) are supplemented by observations of the system as it evolves; this information then changes the probability distribution on the signal, typically

Electronic supplementary material The online version of this chapter (doi: [10.1007/978-3-319-20325-6_2](https://doi.org/10.1007/978-3-319-20325-6_2)) contains supplementary material, which is available to authorized users.

reducing the uncertainty. To describe such situations, we assume that we are given **data**, or **observations**, $y = \{y_j\}_{j \in \mathbb{N}}$ defined as follows. At each discrete time instance, we observe a (possibly nonlinear) function of the signal, with additive noise:

$$y_{j+1} = h(v_{j+1}) + \eta_{j+1}, \quad j \in \mathbb{Z}^+, \quad (2.2)$$

where $h \in C(\mathbb{R}^n, \mathbb{R}^m)$ and $\eta = \{\eta_j\}_{j \in \mathbb{N}}$ is an i.i.d. sequence, independent of (v_0, ξ) , with $\eta_1 \sim N(0, \Gamma)$ and $\Gamma > 0$. The function h is known as the **observation operator**. The objective of **data assimilation** is to determine information about the signal v , given data y . Mathematically, we wish to solve the problem of conditioning the random variable v on the observed data y , or problems closely related to this. Note that we have assumed that both the model noise ξ and the observational noise η are Gaussian; this assumption is made for convenience only, and could be easily generalized.

We will also be interested in the case in which the dynamics is deterministic and (2.1) becomes

$$v_{j+1} = \Psi(v_j), \quad j \in \mathbb{Z}^+, \quad (2.3a)$$

$$v_0 \sim N(m_0, C_0). \quad (2.3b)$$

In this case, which we refer to as **deterministic dynamics**, we are interested in the random variable v_0 , given the observed data y ; note that v_0 determines all subsequent values of the signal v .

Finally, we mention that in many applications, the function Ψ is the solution operator for an ordinary differential equation (ODE) of the form¹

$$\frac{dv}{dt} = f(v), \quad t \in (0, \infty), \quad (2.4a)$$

$$v(0) = v_0. \quad (2.4b)$$

Then, assuming that the solution exists for all $t \geq 0$, there is a one-parameter semigroup of operators $\Psi(\cdot; t)$, parameterized by time $t \geq 0$, with properties defined in (1.10). In this situation, we assume that $\Psi(u) = \Psi(u; \tau)$, i.e., the solution operator over τ time units, where τ is the time between observations; thus we implicitly make the simplifying assumption that the observations are made at equally spaced time points, and note that the state $v_j = v(jh)$ evolves according to (2.3a). We use the notation $\Psi^{(j)}(\cdot)$ to denote the j -fold composition of Ψ with itself. Thus, in the case of continuous-time dynamics, $\Psi(\cdot; j\tau) = \Psi^{(j)}(\cdot)$.

2.2 Guiding Examples

Throughout these notes, we will use the following examples to illustrate the theory and algorithms presented.

Example 2.1. We consider the case of one-dimensional linear dynamics, where

$$\Psi(v) = \lambda v \quad (2.5)$$

for some scalar $\lambda \in \mathbb{R}$. Figure 2.1 compares the behavior of the stochastic dynamics (2.1) and deterministic dynamics (2.3) for the two values $\lambda = 0.5$ and $\lambda = 1.05$. We set $\Sigma = \sigma^2$,

¹ Here the use of $v = \{v(t)\}_{t \geq 0}$ for the solution of this equation should be distinguished from our use of $v = \{v_j\}_{j=0}^\infty$ for the solution of (2.1).

and in both cases, 50 iterations of the map are shown. We observe that the presence of noise does not significantly alter the dynamics of the system for the case $|\lambda| > 1$, since for both the stochastic and deterministic models, $|v_j| \rightarrow \infty$ as $j \rightarrow \infty$. The effects of stochasticity are more pronounced when $|\lambda| < 1$, since in that case, the deterministic map satisfies $v_j \rightarrow 0$, while for the stochastic model, v_j fluctuates randomly around 0.

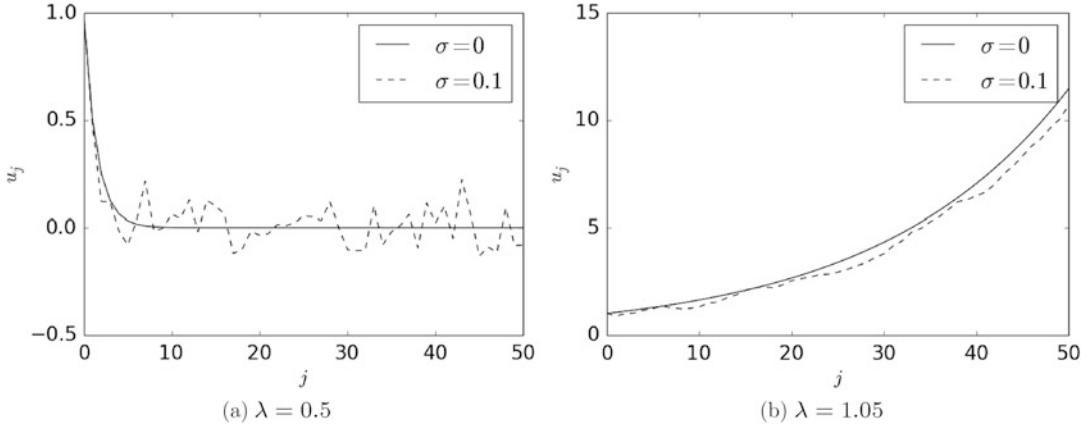


Fig. 2.1: Behavior of (2.1) for Ψ given by (2.5) for different values of λ and $\Sigma = \sigma^2$.

Using (2.1a), together with the linearity of Ψ and the Gaussianity of the noise ξ_j , we obtain

$$\mathbb{E}(v_{j+1}) = \lambda \mathbb{E}(v_j), \quad \mathbb{E}(v_{j+1}^2) = \lambda^2 \mathbb{E}(v_j^2) + \sigma^2.$$

If $|\lambda| > 1$, then the second moment explodes as $j \rightarrow \infty$, as does the modulus of the first moment if $\mathbb{E}(v_0) \neq 0$. On the other hand, if $|\lambda| < 1$, we see (Example 1.15) that $\mathbb{E}(v_j) \rightarrow 0$ and $\mathbb{E}(v_j^2) \rightarrow \sigma_\infty^2$, where

$$\sigma_\infty^2 = \frac{\sigma^2}{1 - \lambda^2}. \quad (2.6)$$

Indeed, since v_0 is Gaussian, the model (2.1a) with linear Ψ and Gaussian noise ξ_j gives rise to a random variable v_j , which is also Gaussian. Thus, from the convergence of the mean and the second moment of v_j , we conclude that v_j converges weakly to the random variable $N(0, \sigma_\infty^2)$. This is an example of ergodicity as expressed in (1.16); the invariant measure μ_∞ is the Gaussian $N(0, \sigma_\infty^2)$, and the density ρ_∞ is the Lebesgue density of this Gaussian. ♠

Example 2.2. Now consider the case of two-dimensional linear dynamics. In this case,

$$\Psi(v) = Av, \quad (2.7)$$

with A a 2×2 matrix of one of the following three forms A_ℓ :

$$A_1 = \begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix}, \quad A_2 = \begin{pmatrix} \lambda & \alpha \\ 0 & \lambda \end{pmatrix}, \quad A_3 = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}.$$

For $\ell = 1, 2$, the behavior of (2.1) for $\Psi(u) = A_\ell u$ can be understood from the analysis underlying Example 2.1, and the behavior is similar, in each coordinate, depending on whether the λ value on the diagonal is less than or greater than 1. However, the picture is more

interesting when we consider the third choice $\Psi(u) = A_3 u$, since in this case, the matrix A_3 has purely imaginary eigenvalues and corresponds to a rotation by $\pi/2$ in the plane; this is illustrated in Figure 2.2a. Addition of noise into the dynamics gives a qualitatively different picture: now the step j to $j + 1$ corresponds to a rotation by $\pi/2$ composed with a random shift of the origin; this is illustrated in Figure 2.2b.

□

♠

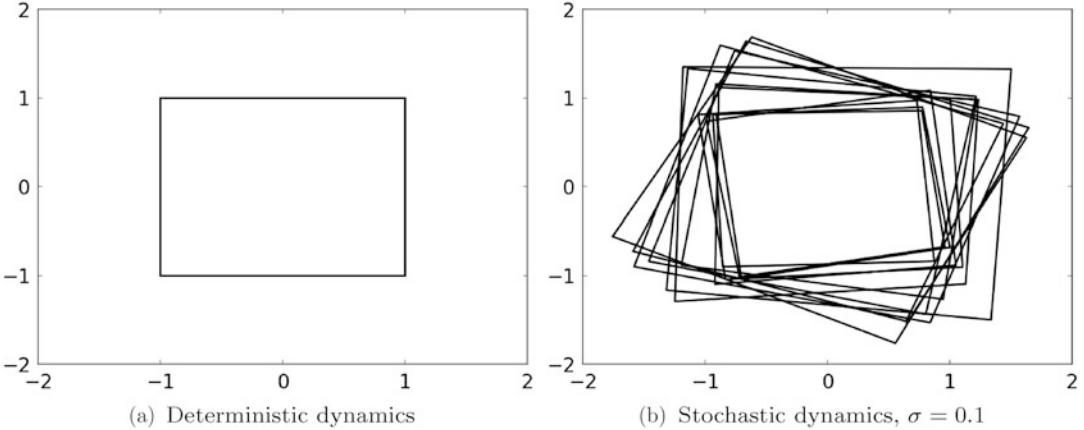


Fig. 2.2: Behavior of (2.1) for Ψ given by (2.7), and $\Sigma = \sigma^2$.

Example 2.3. We now consider our first nonlinear example, namely the one-dimensional dynamics for which

$$\Psi(v) = \alpha \sin v. \quad (2.8)$$

Figure 2.3 illustrates the behavior of (2.1) for this choice of Ψ , and with $\alpha = 2.5$, both for deterministic and stochastic dynamics. In the case of deterministic dynamics, Figure 2.3a, we see that eventually, iterates of the discrete map converge to a period-2 solution. Although only one period-2 solution is seen in this single trajectory, we can deduce that there will be another period-2 solution, related to this one by the symmetry $u \mapsto -u$. This second solution is manifest when we consider stochastic dynamics. Figure 2.3b demonstrates that the inclusion of noise significantly changes the behavior of the system. The signal now exhibits bistable behavior, and within each mode of the behavioral dynamics, vestiges of the period-2 dynamics may be seen: the upper mode of the dynamics is related to the period-2 solution shown in Figure 2.3a, and the lower mode to the period-2 solution found from applying the symmetry $u \mapsto -u$ to obtain a second period-2 solution from that shown in Figure 2.3a.

A good way of visualizing ergodicity is via the *empirical measure* or *histogram* generated by a trajectory of the dynamical system. Equation (1.16) formalizes the idea that the histogram, in the large- J limit, converges to the probability density function of a random variable, independently of the starting point v_0 . Thinking in terms of pdfs of the signal, or functions of the signal, and neglecting time-ordering information is a very useful viewpoint throughout these notes.

Histograms visualize complex dynamical behavior such as that seen in Figure 2.3b by ignoring time-correlation in the signal and simply keeping track of *where* the solution goes as time elapses, but not the *order* in which places are visited. This is illustrated in Figure 2.4a,

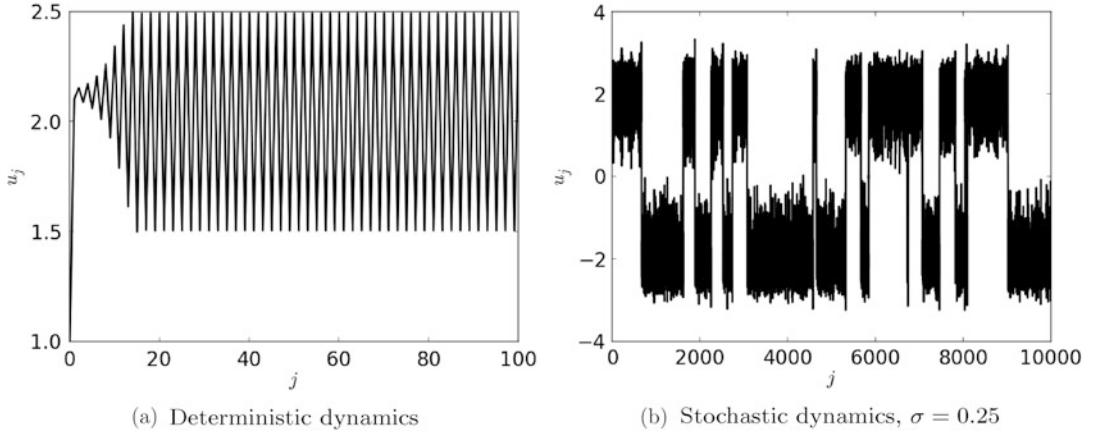


Fig. 2.3: Behavior of (2.1) for Ψ given by (2.8) for $\alpha = 2.5$ and $\Sigma = \sigma^2$; see also p1.m in Section 5.1.1.

where we plot the histogram corresponding to the dynamics shown in Figure 2.3b, but calculated using a simulation of length $J = 10^7$. We observe that the system quickly forgets its initial condition and spends almost equal proportions of time around the positive and negative period-2 solutions of the underlying deterministic map. Figure 2.4a would change very little if the system were started from a different initial condition, reflecting ergodicity of the underlying map. ♠

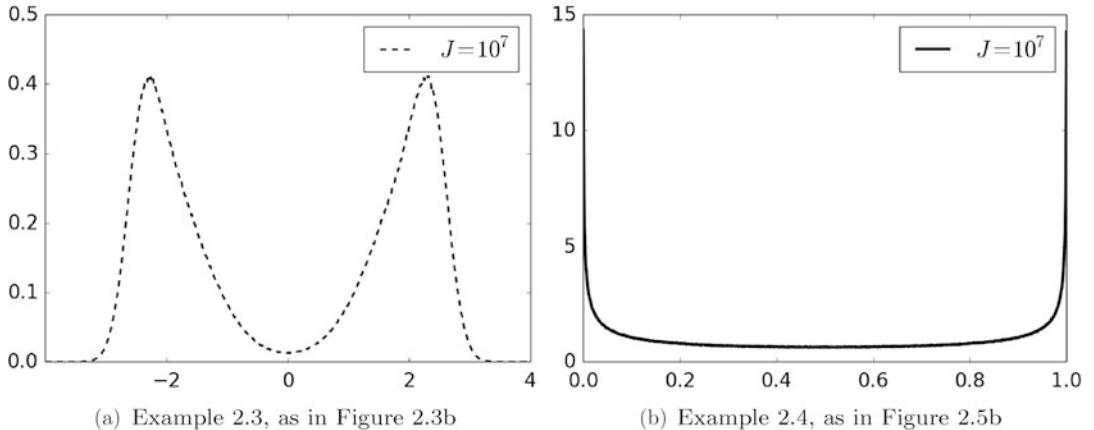


Fig. 2.4: Probability density functions for $v_j, j = 0, \dots, J$, for $J = 10^7$.

Example 2.4. We now consider a second one-dimensional and nonlinear map, for which

$$\Psi(v) = rv(1 - v). \quad (2.9)$$

We consider initial data $v_0 \in [0, 1]$, noting that for $r \in [0, 4]$, the signal will then satisfy $v_j \in [0, 1]$ for all j , in the case of the deterministic dynamics (2.3). We confine our discussion

here to the deterministic case, which can itself exhibit quite rich behavior. In particular, the behavior of (2.3), (2.9) can be seen in Figure 2.5 for the values of $r = 2$ and $r = 4$. These

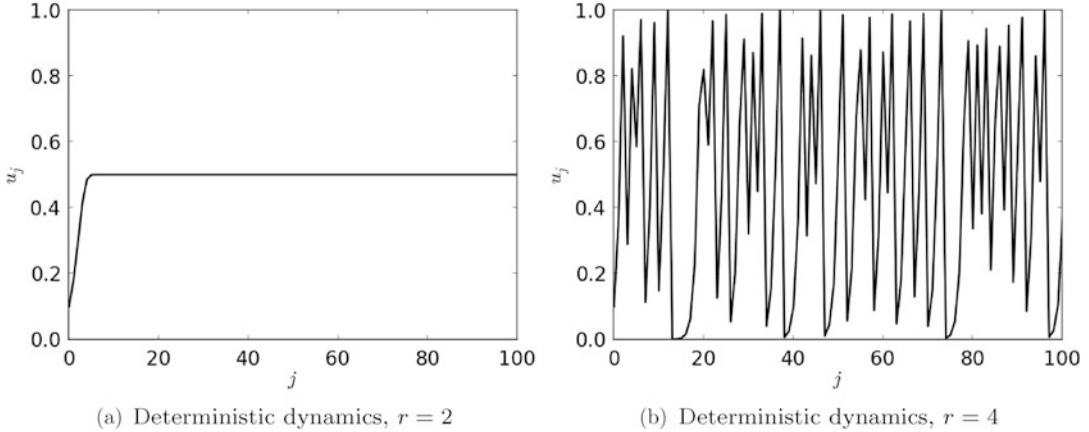


Fig. 2.5: Behavior of (2.1) for Ψ given by (2.9).

values of r have the desirable property that it is possible to determine the signal analytically. For $r = 2$, one obtains

$$v_j = \frac{1}{2} - \frac{1}{2}(1 - 2v_0)^{2^j}, \quad (2.10)$$

which implies that for every value of $v_0 \neq 0, 1$, we have $v_j \rightarrow 1/2$, as we can also see in Figure 2.5a. For $v_0 = 0$, the solution remains at the unstable fixed point 0, while for $v_0 = 1$, the solution maps onto 0 in one step, and then remains there. In the case $r = 4$, the solution is given by

$$v_j = 4 \sin^2(2^j \pi \theta), \quad \text{with } v_0 = 4 \sin^2(\pi \theta). \quad (2.11)$$

This solution can also be expressed in the form

$$v_j = \sin^2(2\pi z_j), \quad (2.12)$$

where

$$z_{j+1} = \begin{cases} 2z_j, & 0 \leq z_j < \frac{1}{2}, \\ 2z_j - 1, & \frac{1}{2} \leq z_j < 1, \end{cases}$$

and using this formula, it is possible to show that this map produces chaotic dynamics for almost all initial conditions. This is illustrated in Figure 2.5b, where we plot the first 100 iterations of the map. In addition, in Figure 2.4b, we plot the pdf using a long trajectory of v_j of length $J = 10^7$, demonstrating the ergodicity of the map. In fact, there is an analytic formula for the steady-state value of the pdf (the invariant density), found as $J \rightarrow \infty$; it is given by

$$\rho(x) = \pi^{-1} x^{-1/2} (1-x)^{-1/2}. \quad (2.13)$$



Example 2.5. Turning now to maps Ψ derived from differential equations, the simplest case is to consider linear autonomous dynamical systems of the form

$$\frac{dv}{dt} = Lv, \quad (2.14a)$$

$$v(0) = v_0. \quad (2.14b)$$

Then $\Psi(u) = Au$ with $A = \exp(L\tau)$. ♠

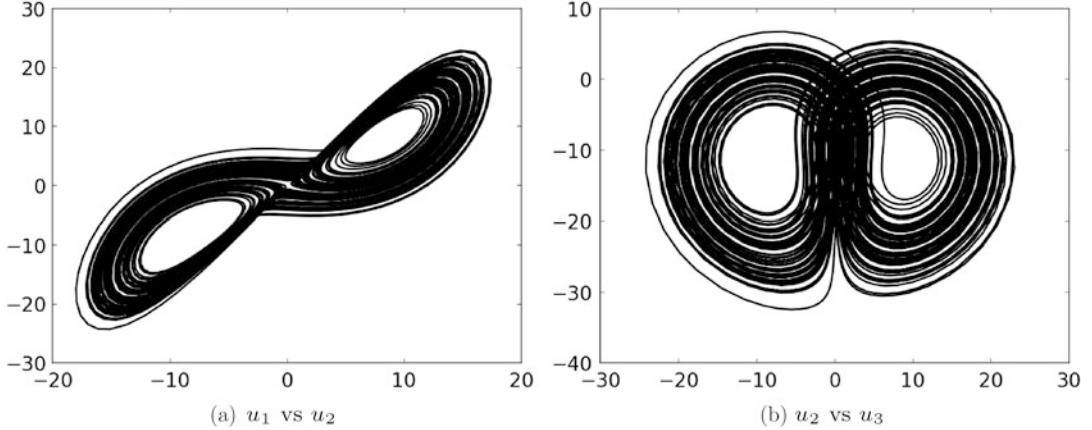


Fig. 2.6: Projection of the Lorenz '63 attractor onto two different pairs of coordinates.

Example 2.6. The Lorenz '63 model is perhaps the simplest continuous-time system to exhibit sensitivity to initial conditions and chaos. It is a system of three coupled nonlinear ordinary differential equations whose solution $v \in \mathbb{R}^3$, where $v = (v_1, v_2, v_3)$, satisfies²

$$\frac{dv_1}{dt} = a(v_2 - v_1), \quad (2.15a)$$

$$\frac{dv_2}{dt} = -av_1 - v_2 - v_1v_3, \quad (2.15b)$$

$$\frac{dv_3}{dt} = v_1v_2 - bv_3 - b(r + a). \quad (2.15c)$$

Note that we have employed a coordinate system in which the origin in the original version of the equations proposed by Lorenz is shifted. In the coordinate system that we employ here, we have equation (2.4) with vector field f satisfying

$$\langle f(v), v \rangle \leq \alpha - \beta|v|^2 \quad (2.16)$$

for some $\alpha, \beta > 0$. As demonstrated in Example 1.22, this implies the existence of an absorbing set:

$$\limsup_{t \rightarrow \infty} |v(t)|^2 < R \quad (2.17)$$

for every $R > \alpha/\beta$. Mapping the ball $B(0, R)$ forward under the dynamics gives the global attractor (see Definition 1.23) for the dynamics. In Figure 2.6, we visualize this attractor, projected onto two different pairs of coordinates at the classical parameter values $(a, b, r) = (10, \frac{8}{3}, 28)$.

² Here the index denotes components of the solution, not discrete time.

Throughout these notes, we will use the classical parameter values $(a, b, r) = (10, \frac{8}{3}, 28)$ in all of our numerical experiments; at these values, the system is chaotic, and it exhibits sensitive dependence with respect to the initial condition. A trajectory of v_1 versus time can be found in Figure 2.7a, and in Figure 2.7b, we illustrate the evolution of a small perturbation to the initial condition that generated Figure 2.7a; to be explicit, we plot the evolution of the error in the Euclidean norm $|\cdot|$ for an initial perturbation of magnitude 10^{-4} . Figure 2.6 suggests that the measure μ_∞ is supported on a strange set with Lebesgue measure zero, and this is indeed the case; for this example, there is no Lebesgue density ρ_∞ for the invariant measure, reflecting the fact that the attractor has a fractal dimension less than three, the dimension of the space where the dynamical system lies.

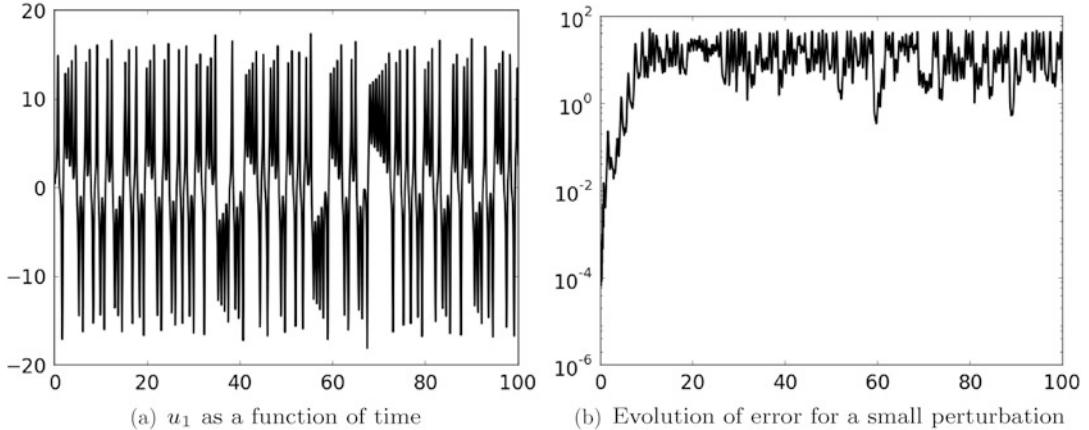


Fig. 2.7: Dynamics of the Lorenz '63 model in the chaotic regime $(a, b, r) = (10, \frac{8}{3}, 28)$.

Example 2.7. The Lorenz-96 model is a simple dynamical system, of tunable dimension, that was designed as a caricature of the dynamics of Rossby waves in atmospheric dynamics. The equations have a periodic “ring” formulation and take the form³

$$\frac{dv_k}{dt} = v_{k-1}(v_{k+1} - v_{k-2}) - v_k + F, \quad k \in \{1, \dots, K\}, \quad (2.18a)$$

$$v_0 = v_K, \quad v_{K+1} = v_1, \quad v_{-1} = v_{K-1}. \quad (2.18b)$$

Equation (2.18) satisfies the same dissipativity property (2.16) satisfied by the Lorenz '63 model, for appropriate choice of $\alpha, \beta > 0$, and hence also satisfies the absorbing ball property (2.17), thus having a global attractor (see Definition 1.23).

In Figure 2.8a, we plot a trajectory of v_1 versus time for $F = 8$ and $K = 40$. Furthermore, as we did in the case of the Lorenz '63 model, we also show the evolution of the Euclidean norm of the error $|\cdot|$ for an initial perturbation of magnitude 10^{-4} ; this is displayed in Figure 2.8b and clearly demonstrates sensitive dependence on initial conditions. We visualize the attractor, projected onto two different pairs of coordinates, in Figure 2.9.



³ Again, here the index denotes components of the solution, not discrete time.

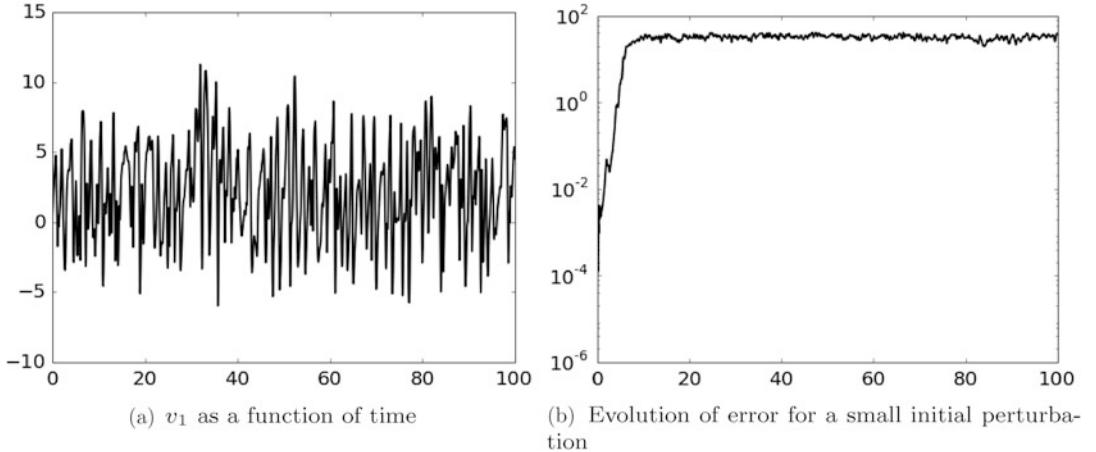


Fig. 2.8: Dynamics of the Lorenz-96 model in the chaotic regime $(F, K) = (8, 40)$.

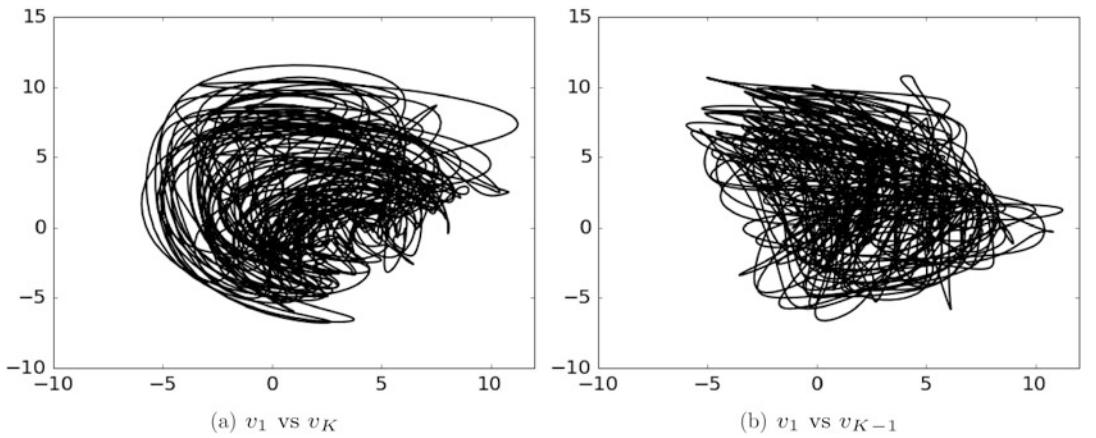


Fig. 2.9: Projection of the Lorenz-96 attractor onto two different pairs of coordinates.

2.3 Smoothing Problem

2.3.1. Probabilistic Formulation of Data Assimilation

Together, (2.1) and (2.2) provide a probabilistic model for the jointly varying random variable (v, y) . In the case of deterministic dynamics, (2.3) and (2.2) provide a probabilistic model for the jointly varying random variable (v_0, y) . Thus in both cases, we have a random variable (u, y) , with $u = v$ (respectively $u = v_0$) in the stochastic (respectively deterministic) case. Our aim is to discover information about the signal v in the stochastic case, or v_0 in the deterministic case, from observation of a single instance of the data y . The natural probabilistic approach to this problem is to try to find the probability measure describing the random variable u given y , denoted by $u|y$. This constitutes the Bayesian formulation of the problem of determining information about the signal arising in a noisy dynamical model based on noisy observations of that signal. We will refer to the conditioned random variable $u|y$ in the

case of either the stochastic dynamics or the deterministic dynamics as the **smoothing distribution**. It is a random variable that contains all the probabilistic information about the signal, given our observations. The key concept that drives this approach is Bayes's formula from Section 1.1.4, which we use repeatedly in what follows.

2.3.2. Stochastic Dynamics

We wish to find the signal v from (2.1) from a single instance of data y given by (2.2). To be more precise, we wish to condition the signal on a discrete time interval $\mathbb{J}_0 = \{0, \dots, J\}$, given data on the discrete time interval $\mathbb{J} = \{1, \dots, J\}$; we refer to \mathbb{J}_0 as the data assimilation window. We define $v = \{v_j\}_{j \in \mathbb{J}_0}$, $y = \{y_j\}_{j \in \mathbb{J}}$, $\xi = \{\xi_j\}_{j \in \mathbb{J}_0}$, and $\eta = \{\eta_j\}_{j \in \mathbb{J}}$. The smoothing distribution here is the distribution of the conditioned random variable $v|y$. Recall that we have assumed that v_0 , ξ , and η are mutually independent random variables. With this fact in hand, we may apply Bayes's formula to find the pdf $\mathbb{P}(v|y)$.

Prior. The prior on v is specified by (2.1), together with the independence of u and ξ and the i.i.d. structure of ξ . First note that using (1.5) and the i.i.d. structure of ξ in turn, we obtain

$$\begin{aligned}\mathbb{P}(v) &= \mathbb{P}(v_J, v_{J-1}, \dots, v_0) \\ &= \mathbb{P}(v_J|v_{J-1}, \dots, v_0)\mathbb{P}(v_{J-1}, \dots, v_0) \\ &= \mathbb{P}(v_J|v_{J-1})\mathbb{P}(v_{J-1}, \dots, v_0).\end{aligned}$$

Proceeding inductively gives

$$\mathbb{P}(v) = \prod_{j=0}^{J-1} \mathbb{P}(v_{j+1}|v_j)\mathbb{P}(v_0).$$

Now

$$\mathbb{P}(v_0) \propto \exp\left(-\frac{1}{2}|C_0^{-\frac{1}{2}}(v_0 - m_0)|^2\right),$$

while

$$\mathbb{P}(v_{j+1}|v_j) \propto \exp\left(-\frac{1}{2}\left|\Sigma^{-\frac{1}{2}}(v_{j+1} - \Psi(v_j))\right|^2\right).$$

The probability distribution $\mathbb{P}(v)$ that we now write down is *not* Gaussian, but the distribution on the initial condition $\mathbb{P}(v_0)$, and the conditional distributions $\mathbb{P}(v_{j+1}|v_j)$, are all Gaussian, making the explicit calculations above straightforward.

Combining the preceding information, we obtain

$$\mathbb{P}(v) \propto \exp(-\mathbf{J}(v)),$$

where

$$\mathbf{J}(v) := \frac{1}{2}|C_0^{-\frac{1}{2}}(v_0 - m_0)|^2 + \sum_{j=0}^{J-1} \frac{1}{2}\left|\Sigma^{-\frac{1}{2}}(v_{j+1} - \Psi(v_j))\right|^2 \quad (2.19a)$$

$$= \frac{1}{2}|v_0 - m_0|_{C_0}^2 + \sum_{j=0}^{J-1} \frac{1}{2}\left|v_{j+1} - \Psi(v_j)\right|_\Sigma^2. \quad (2.19b)$$

The pdf $\mathbb{P}(v) = \rho_0(v)$ proportional to $\exp(-\mathbf{J}(v))$ determines a prior measure μ_0 on $\mathbb{R}^{|\mathbb{J}_0| \times n}$. The fact that the probability is not in general Gaussian follows from the fact that Ψ is not in general linear.

Likelihood. The likelihood of the data $y|v$ is determined as follows. It is a (Gaussian) probability distribution on $\mathbb{R}^{|\mathbb{J}| \times m}$, with pdf $\mathbb{P}(y|v)$ proportional to $\exp(-\Phi(v; y))$, where

$$\Phi(v; y) = \sum_{j=0}^{J-1} \frac{1}{2} |y_{j+1} - h(v_{j+1})|_\Gamma^2. \quad (2.20)$$

To see this, note that because of the i.i.d. nature of the sequence η , it follows that

$$\begin{aligned} \mathbb{P}(y|v) &= \prod_{j=0}^{J-1} \mathbb{P}(y_{j+1}|v) \\ &= \prod_{j=0}^{J-1} \mathbb{P}(y_{j+1}|v_{j+1}) \\ &\propto \prod_{j=0}^{J-1} \exp\left(-\frac{1}{2} |\Gamma^{-\frac{1}{2}}(y_{j+1} - h(v_{j+1}))|^2\right) \\ &= \exp(-\Phi(v; y)). \end{aligned}$$

In the applied literature, m_0 and C_0 are often referred to as the **background mean** and **background covariance** respectively; we refer to Φ as the **model–data misfit** functional.

Using Bayes's formula (1.7), we can combine the prior and the likelihood to determine the posterior distribution, that is, the smoothing distribution, on $v|y$. We denote the measure with this distribution by μ .

Theorem 2.8. *The posterior smoothing distribution on $v|y$ for the stochastic dynamics model (2.1), (2.2) is a probability measure μ on $\mathbb{R}^{|\mathbb{J}_0| \times n}$ with pdf $\mathbb{P}(v|y) = \rho(v)$ proportional to $\exp(-\mathbf{l}(v; y))$, where*

$$\mathbf{l}(v; y) = \mathbf{J}(v) + \Phi(v; y). \quad (2.21)$$

Proof Bayes's formula (1.7) gives us

$$\mathbb{P}(v|y) = \frac{\mathbb{P}(y|v)\mathbb{P}(v)}{\mathbb{P}(y)}.$$

Thus, ignoring constants of proportionality that depend only on y , we have

$$\begin{aligned} \mathbb{P}(v|y) &\propto \mathbb{P}(y|v)\mathbb{P}(v_0) \\ &\propto \exp(-\Phi(v; y)) \exp(-\mathbf{J}(v)) \\ &= \exp(-\mathbf{l}(v; y)). \end{aligned}$$

□

Note that although the preceding calculations required only knowledge of the pdfs of Gaussian distributions, the resulting posterior distribution is non-Gaussian in general, unless Ψ and h are linear. This is because unless Ψ and h are linear, $\mathbf{l}(\cdot; y)$ is not quadratic. We refer to \mathbf{l} as the negative log-posterior. It will be helpful later to note that

$$\frac{\rho(v)}{\rho_0(v)} \propto \exp(-\Phi(v; y)). \quad (2.22)$$

2.3.3. Reformulation of Stochastic Dynamics

For the development of algorithms to probe the posterior distribution, the following reformulation of the stochastic dynamics problem can be very useful. For this, we define the vector $\xi = (v_0, \xi_0, \xi_1, \dots, \xi_{J-1}) \in \mathbb{R}^{|\mathbb{J}_0|n}$. The following lemma is key to what follows.

Lemma 2.9. Define the mapping $G : \mathbb{R}^{|\mathbb{J}_0| \times n} \mapsto \mathbb{R}^{|\mathbb{J}_0| \times n}$ by

$$G_j(v_0, \xi_0, \xi_1, \dots, \xi_{J-1}) = v_j, \quad j = 0, \dots, J,$$

where v_j is determined by (2.1). Then this mapping is invertible. Furthermore, if $\Psi \equiv 0$, then G is the identity mapping.

Proof In words, the mapping G takes the initial condition and noise into the signal. Invertibility requires determination of the initial condition and the noise from the signal. From the signal, we may compute the noise as follows, noting that of course, the initial condition is specified, and that then we have

$$\xi_j = v_{j+1} - \Psi(v_j), \quad j = 0, \dots, J-1.$$

The fact that G becomes the identity mapping when $\Psi \equiv 0$ follows directly from (2.1) by inspection. \square

We may thus consider the smoothing problem as finding the probability distribution of ξ , as defined prior to the lemma, given data y , with y as defined in Section 2.3.2. Furthermore, we have, using the notion of pushforward,

$$\mathbb{P}(v|y) = G \star \mathbb{P}(\xi|y), \quad \mathbb{P}(\xi|y) = G^{-1} \star \mathbb{P}(v|y). \quad (2.23)$$

These formulas mean that it is easy to move between the two measures: samples from one can be converted into samples from the other simply by applying G or G^{-1} . This means that algorithms can be applied, for example, to generate samples from $\xi|y$ and then convert them into samples from $v|y$. We will use this later on. In order to use this idea, it will be helpful to have an explicit expression for the pdf of $\xi|y$. We now find such an expression.

To begin, we introduce the measure ϑ_0 with density π_0 found from μ_0 and ρ_0 in the case $\Psi \equiv 0$. Thus

$$\pi_0(v) \propto \exp \left(-\frac{1}{2} \left| C_0^{-\frac{1}{2}}(v_0 - m_0) \right|^2 - \sum_{j=0}^{J-1} \frac{1}{2} |\Sigma^{-\frac{1}{2}} v_{j+1}|^2 \right) \quad (2.24a)$$

$$\propto \exp \left(-\frac{1}{2} |v_0 - m_0|_{C_0}^2 - \sum_{j=0}^{J-1} \frac{1}{2} |v_{j+1}|_{\Sigma}^2 \right), \quad (2.24b)$$

and hence ϑ_0 is a Gaussian measure, independent in each component v_j for $j = 0, \dots, J$. By Lemma 2.9, we also deduce that measure ϑ_0 with density π_0 is the prior on ξ as defined above:

$$\pi_0(\xi) \propto \exp \left(-\frac{1}{2} |v_0 - m_0|_{C_0}^2 - \sum_{j=0}^{J-1} \frac{1}{2} |\xi_j|_{\Sigma}^2 \right). \quad (2.25)$$

We now compute the likelihood of $y|\xi$. For this, we define

$$\mathcal{G}_j(\xi) = h(G_j(\xi)) \quad (2.26)$$

and note that we may then concatenate the data and write

$$y = \mathcal{G}(\xi) + \eta, \quad (2.27)$$

where $\eta = (\eta_1, \dots, \eta_J)$ is the Gaussian random variable $N(0, \Gamma_J)$, where Γ_J is a block diagonal $nJ \times nJ$ matrix with $n \times n$ diagonal blocks Γ . It follows that the likelihood is determined by $\mathbb{P}(y|\xi) = N(\mathcal{G}(\xi), \Gamma_J)$. Applying Bayes's formula from (1.7) to find the pdf for $\xi|y$, we obtain the posterior ϑ on $\xi|y$, as summarized in the following theorem.

Theorem 2.10. *The posterior smoothing distribution on $\xi|y$ for the stochastic dynamics model (2.1), (2.2) is a probability measure ϑ on $\mathbb{R}^{|\mathbb{J}_0| \times n}$ with pdf $\mathbb{P}(\xi|y) = \pi(\xi)$ proportional to $\exp(-\mathbf{l}_r(\xi; y))$, where*

$$\begin{aligned} \mathbf{l}_r(\xi; y) &= \mathbf{J}_r(\xi) + \Phi_r(\xi; y), \\ \Phi_r(\xi; y) &:= \frac{1}{2} |(y - \mathcal{G}(\xi))|_{\Gamma_J}^2, \end{aligned} \quad (2.28)$$

and

$$\mathbf{J}_r(\xi) := \frac{1}{2} |v_0 - m_0|_{C_0}^2 + \sum_{j=0}^{J-1} \frac{1}{2} |\xi_j|_{\Sigma}^2.$$

We refer to \mathbf{l}_r as the negative log-posterior.

2.3.4. Deterministic Dynamics

It is also of interest to study the posterior distribution on the initial condition in the case that the model dynamics contains no noise and is given by (2.3); this we now do. Recall that $\Psi^{(j)}(\cdot)$ denotes the j -fold composition of $\Psi(\cdot)$ with itself. In the following, we sometimes refer to \mathbf{J}_{det} as the **background** penalization, and m_0 and C_0 as the background mean and covariance; we refer to Φ_{det} as the **model–data misfit** functional.

Theorem 2.11. *The posterior smoothing distribution on $v_0|y$ for the deterministic dynamics model (2.3), (2.2) is a probability measure ν on \mathbb{R}^n with density $\mathbb{P}(v_0|y) = \varrho(v_0)$ proportional to $\exp(-\mathbf{l}_{\text{det}}(v_0; y))$, where*

$$\mathbf{l}_{\text{det}}(v_0; y) = \mathbf{J}_{\text{det}}(v_0) + \Phi_{\text{det}}(v_0; y), \quad (2.29a)$$

$$\mathbf{J}_{\text{det}}(v_0) = \frac{1}{2} |v_0 - m_0|_{C_0}^2, \quad (2.29b)$$

$$\Phi_{\text{det}}(v_0; y) = \sum_{j=0}^{J-1} \frac{1}{2} |y_{j+1} - h(\Psi^{(j+1)}(v_0))|_{\Gamma}^2. \quad (2.29c)$$

Proof We again use Bayes's rule, which states that

$$\mathbb{P}(v_0|y) = \frac{\mathbb{P}(y|v_0)\mathbb{P}(v_0)}{\mathbb{P}(y)}.$$

Thus, ignoring constants of proportionality that depend only on y , we have

$$\begin{aligned}\mathbb{P}(v_0|y) &\propto \mathbb{P}(y|v_0)\mathbb{P}(v_0) \\ &\propto \exp(-\Phi_{\text{det}}(v_0; y)) \exp\left(-\frac{1}{2}|v_0 - m_0|_{C_0}^2\right) \\ &= \exp(-\mathbf{l}_{\text{det}}(v_0; y)).\end{aligned}$$

Here we have used the fact that $\mathbb{P}(y|v_0)$ is proportional to $\exp(-\Phi_{\text{det}}(v_0; y))$; this follows from the fact that $y_j|v_0$ form an i.i.d. sequence of Gaussian random variables $N(h(v_j), \Gamma)$ with $v_j = \Psi^{(j)}(v_0)$. \square

We refer to \mathbf{l}_{det} as the negative log-posterior.

2.4 Filtering Problem

The smoothing problem considered in the previous section involves, potentially, conditioning v_j on data y_k with $k > j$. Such conditioning can be performed only *offline* and is of no use in *online* scenarios in which we want to determine information on the state of the signal *now*, hence using only data from the past up to the present. To study this situation, let $Y_j = \{y_l\}_{l=1}^j$ denote the accumulated data up to time j . **Filtering** is concerned with determining $\mathbb{P}(v_j|Y_j)$, the pdf associated with the probability measure on the random variable $v_j|Y_j$; in particular, filtering is concerned with the sequential updating of this pdf as the index j is incremented. This update is defined by the following procedure, which provides a prescription for computing $\mathbb{P}(v_{j+1}|Y_{j+1})$ from $\mathbb{P}(v_j|Y_j)$ via two steps: **prediction**, which computes the mapping $\mathbb{P}(v_j|Y_j) \mapsto \mathbb{P}(v_{j+1}|Y_j)$, and **analysis**, which computes $\mathbb{P}(v_{j+1}|Y_j) \mapsto \mathbb{P}(v_{j+1}|Y_{j+1})$ by application of Bayes's formula.

Prediction. Note that $\mathbb{P}(v_{j+1}|Y_j, v_j) = \mathbb{P}(v_{j+1}|v_j)$, because Y_j contains noisy and indirect information about v_j and cannot improve upon perfect knowledge of the variable v_j . Thus, by (1.5), we deduce that

$$\mathbb{P}(v_{j+1}|Y_j) = \int_{\mathbb{R}^n} \mathbb{P}(v_{j+1}|Y_j, v_j) \mathbb{P}(v_j|Y_j) dv_j \quad (2.30a)$$

$$= \int_{\mathbb{R}^n} \mathbb{P}(v_{j+1}|v_j) \mathbb{P}(v_j|Y_j) dv_j. \quad (2.30b)$$

Note that since the forward model equation (2.1) determines $\mathbb{P}(v_{j+1}|v_j)$, this prediction step provides the map from $\mathbb{P}(v_j|Y_j)$ to $\mathbb{P}(v_{j+1}|Y_j)$. This prediction step simplifies in the case of deterministic dynamics (2.3); in this case, it simply corresponds to computing the pushforward of $\mathbb{P}(v_j|Y_j)$ under the map Ψ .

Analysis. Note that $\mathbb{P}(y_{j+1}|v_{j+1}, Y_j) = \mathbb{P}(y_{j+1}|v_{j+1})$, because Y_j contains noisy and indirect information about v_j and cannot improve upon perfect knowledge of the variable v_{j+1} . Thus, using Bayes's formula (1.7), we deduce that

$$\begin{aligned}\mathbb{P}(v_{j+1}|Y_{j+1}) &= \mathbb{P}(v_{j+1}|Y_j, y_{j+1}) \\ &= \frac{\mathbb{P}(y_{j+1}|v_{j+1}, Y_j) \mathbb{P}(v_{j+1}|Y_j)}{\mathbb{P}(y_{j+1}|Y_j)} \\ &= \frac{\mathbb{P}(y_{j+1}|v_{j+1}) \mathbb{P}(v_{j+1}|Y_j)}{\mathbb{P}(y_{j+1}|Y_j)}.\end{aligned} \quad (2.31)$$

Since the observation equation (2.2) determines $\mathbb{P}(y_{j+1}|v_{j+1})$, this analysis step provides a map from $\mathbb{P}(v_{j+1}|Y_j)$ to $\mathbb{P}(v_{j+1}|Y_{j+1})$.

Filtering Update. Together, then, the prediction and analysis step provide a mapping from $\mathbb{P}(v_j|Y_j)$ to $\mathbb{P}(v_{j+1}|Y_{j+1})$. Indeed, if we let μ_j denote the probability measure on \mathbb{R}^n corresponding to the density $\mathbb{P}(v_j|Y_j)$, and let $\hat{\mu}_{j+1}$ be the probability measure on \mathbb{R}^n corresponding to the density $\mathbb{P}(v_{j+1}|Y_j)$, then the prediction step maps μ_j to $\hat{\mu}_{j+1}$, while the analysis step maps $\hat{\mu}_{j+1}$ to μ_{j+1} . However, there is, in general, no easily usable closed-form expression for the density of μ_j , namely $\mathbb{P}(v_j|Y_j)$. Nevertheless, formulas (2.30), (2.31) form the starting point for numerous algorithms to approximate $\mathbb{P}(v_j|Y_j)$. In terms of analyzing the particle filter, it is helpful conceptually to write the prediction and analysis steps as

$$\hat{\mu}_{j+1} = P\mu_j \quad \mu_{j+1} = L_j \hat{\mu}_{j+1}. \quad (2.32)$$

Note that P does not depend on j , since the same Markov process governs the prediction step at each j ; however, L_j depends on j , because the likelihood sees different data at each j . Furthermore, the formula $\hat{\mu}_{j+1} = P\mu_j$ summarizes (2.30), while $\mu_{j+1} = L_j \hat{\mu}_{j+1}$ summarizes (2.31). Note that P is a linear mapping, while L_j is nonlinear; this issue is discussed in Sections 1.4.1 and 1.4.3 at the level of pdfs.

2.5 Filtering and Smoothing are Related

The filtering and smoothing approaches to determining the signal from the data are distinct but related. They are related by the fact that in both cases, the solutions computed at the end of any specified time interval are conditioned on the same data and must hence coincide; this is made precise in the following.

Theorem 2.12. *Let $\mathbb{P}(v|y)$ denote the smoothing distribution on the discrete time interval $j \in \mathbb{J}_0$, and $\mathbb{P}(v_J|Y_J)$ the filtering distribution at time $j = J$ for the stochastic dynamics model (2.1). Then the marginal of the smoothing distribution on v_J is the same as the filtering distribution at time J :*

$$\int \mathbb{P}(v|y) dv_0 dv_1 \dots dv_{J-1} = \mathbb{P}(v_J|Y_J).$$

Proof Note that $y = Y_J$. Since $v = (v_0, \dots, v_{J-1}, v_J)$, the result follows trivially. \square

Remark 2.13. *Note that the marginal of the smoothing distribution on say v_j , $j < J$, is not equal to the filter $\mathbb{P}(v_j|Y_j)$. This is because the smoother induces a distribution on v_j that is influenced by the entire data set $Y_J = y = \{y_l\}_{l \in \mathbb{J}}$; in contrast, the filter at j involves only the data $Y_j = \{y_l\}_{l \in \{1, \dots, j\}}$.* ♠

It is also interesting to mention the relationship between filtering and smoothing in the case of noise-free dynamics. In this case, the filtering distribution $\mathbb{P}(v_j|Y_j)$ is simply found as the pushforward of the smoothing distribution on $\mathbb{P}(v_0|Y_j)$ under $\Psi^{(j)}$, that is, under j applications of Ψ .

Theorem 2.14. *Let $\mathbb{P}(v_0|y)$ denote the smoothing distribution on the discrete time interval $j \in \mathbb{J}_0$, and $\mathbb{P}(v_J|Y_J)$ the filtering distribution at time $j = J$ for the deterministic dynamics model (2.3). Then the pushforward of the smoothing distribution on v_0 under $\Psi^{(J)}$ is the same as the filtering distribution at time J :*

$$\Psi^{(J)} \star \mathbb{P}(v_0|Y_J) = \mathbb{P}(v_J|Y_J).$$

2.6 Well-Posedness

Well-posedness of a mathematical problem refers, generally, to the existence of a unique solution that depends continuously on the parameters defining the problem. We have shown, for both filtering and smoothing, how to construct a uniquely defined probabilistic solution to the problem of determining the signal given the data. In this setting, it is natural to consider well-posedness with respect to the data itself. Thus we now investigate the continuous dependence of the probabilistic solution on the observed data; indeed, we will show Lipschitz dependence. To this end, we need probability metrics, as introduced in Section 1.3.

As we do throughout these notes, we perform all calculations using the existence of everywhere positive Lebesgue densities for our measures. We let μ_0 denote the prior measure on v for the smoothing problem arising in stochastic dynamics, as defined by (2.1). Then μ and μ' denote the posterior measures resulting from two different instances of the data, y and y' respectively. Let ρ_0 , ρ , and ρ' denote the Lebesgue densities on μ_0 , μ , and μ' respectively. Then for J and Φ as defined in (2.19) and (2.20),

$$\rho_0(v) = \frac{1}{Z_0} \exp(-J(v)), \quad (2.33a)$$

$$\rho(v) = \frac{1}{Z} \exp(-J(v) - \Phi(v; y)), \quad (2.33b)$$

$$\rho'(v) = \frac{1}{Z'} \exp(-J(v) - \Phi(v; y')), \quad (2.33c)$$

where

$$Z_0 = \int \exp(-J(v)) dv, \quad (2.34a)$$

$$Z = \int \exp(-J(v) - \Phi(v; y)) dv, \quad (2.34b)$$

$$Z' = \int \exp(-J(v) - \Phi(v; y')) dv. \quad (2.34c)$$

Here, and in the proofs that follow in this section, all integrals are over $\mathbb{R}^{|\mathbb{J}_0| \times n}$ (or in the case of the deterministic dynamics model at the end of the section, over \mathbb{R}^n). Note that $|\mathbb{J}_0|$ is the cardinality of the set \mathbb{J}_0 and is hence equal to $J + 1$. To this end, we note explicitly that (2.33a) implies that

$$\exp(-J(v)) dv = Z_0 \rho_0(v) dv = Z_0 \mu_0(dv), \quad (2.35)$$

indicating that integrals weighted by $\exp(-J(v))$ may be rewritten as expectations with respect to μ_0 . We use the identities (2.33), (2.34), and (2.35) repeatedly in what follows to express all integrals as expectations with respect to the measure μ_0 . In particular, the assumptions that we make for the subsequent theorems and corollaries in this section are all expressed in terms of expectations under μ_0 (or under ν_0 for the deterministic dynamics problem considered at the end of the section). This is convenient, because it relates to the unconditioned problem of stochastic dynamics for v in the absence of any data, and may thus be checked once and for all, independently of the particular data set y or y' that are used to condition v and obtain μ and μ' .

We assume throughout what follows that y, y' are both contained in a ball of radius r in the Euclidean norm on $\mathbb{R}^{|\mathbb{J}| \times n}$. Again, $|\mathbb{J}|$ is the cardinality of the set \mathbb{J} and is hence

equal to J . We also note that Z_0 is bounded from above independently of r , because ρ_0 is the density associated with the probability measure μ_0 , which is therefore normalizable, and this measure is independent of the data. It also follows that $Z \leq Z_0$, $Z' \leq Z_0$ using (2.35) in (2.34b), (2.34c), together with the fact that $\Phi(\cdot; y)$ is a positive function. Furthermore, if we assume that

$$\mathbf{v} := \sum_{j \in \mathbb{J}} (1 + |h(v_j)|^2) \quad (2.36)$$

satisfies $\mathbb{E}^{\mu_0} \mathbf{v} < \infty$, then both Z and Z' are positive with common lower bound depending only on r , as we now demonstrate. It is sufficient to prove the result for Z , which we now do. In the following, and in the proofs that follow, K denotes a generic constant, which may depend on r and J but not on the solution sequence v , and which may change from instance to instance. Note first that by (2.34), (2.35),

$$\frac{Z}{Z_0} = \int \exp(-\Phi(v; y)) \rho_0(v) dv \geq \int \exp(-K \mathbf{v}) \rho_0(v) dv.$$

Since $\mathbb{E}^{\mu_0} \mathbf{v} < \infty$, we deduce from (1.2) that for R sufficiently large,

$$\begin{aligned} \frac{Z}{Z_0} &\geq \exp(-KR) \int_{|\mathbf{v}| < R} \rho_0(v) dv = \exp(-KR) \mathbb{P}^{\mu_0}(|\mathbf{v}| < R) \\ &\geq \exp(-KR) (1 - R^{-1} \mathbb{E}^{\mu_0} \mathbf{v}). \end{aligned}$$

Since K depends on y, y' only through r , we deduce that by choosing R sufficiently large, we have found lower bounds on Z, Z' that depend on y, y' only through r .

Finally, we note that since all norms are equivalent on finite-dimensional spaces, there is constant K such that

$$\left(\sum_{j=0}^{J-1} |y_{j+1} - y'_{j+1}|_T^2 \right)^{\frac{1}{2}} \leq K|y - y'|. \quad (2.37)$$

The following theorem then shows that the posterior measure is in fact Lipschitz continuous, in the Hellinger metric, with respect to the data.

Theorem 2.15. *Consider the smoothing problem arising from the stochastic dynamics model (2.1), resulting in the posterior probability distributions μ and μ' associated with two different data sets y and y' . Assume that $\mathbb{E}^{\mu_0} \mathbf{v} < \infty$, where \mathbf{v} is given by (2.36). Then there exists $c = c(r)$ such that for all $|y|, |y'| \leq r$,*

$$d_{\text{Hell}}(\mu, \mu') \leq c|y - y'|.$$

Proof We have, by (2.33), (2.35),

$$\begin{aligned} d_{\text{Hell}}(\mu, \mu')^2 &= \frac{1}{2} \int |\sqrt{\rho(v)} - \sqrt{\rho'(v)}|^2 dv \\ &= \frac{1}{2} \int Z_0 \left| \frac{1}{\sqrt{Z}} e^{-\frac{1}{2}\Phi(v; y)} - \frac{1}{\sqrt{Z'}} e^{-\frac{1}{2}\Phi(v; y')} \right|^2 \rho_0(v) dv \\ &\leq I_1 + I_2, \end{aligned}$$

where

$$I_1 = Z_0 \int \frac{1}{Z} \left| e^{-\frac{1}{2}\Phi(v;y)} - e^{-\frac{1}{2}\Phi(v;y')} \right|^2 \rho_0(v) dv,$$

and using (2.33) and (2.34),

$$\begin{aligned} I_2 &= Z_0 \left| \frac{1}{\sqrt{Z}} - \frac{1}{\sqrt{Z'}} \right|^2 \int e^{-\Phi(v;y')} \rho_0(v) dv \\ &= Z' \left| \frac{1}{\sqrt{Z}} - \frac{1}{\sqrt{Z'}} \right|^2. \end{aligned}$$

We estimate I_2 first. Since as shown before the theorem, Z, Z' are bounded below by a positive constant depending only on r , we have

$$I_2 = \frac{1}{Z} |\sqrt{Z} - \sqrt{Z'}|^2 = \frac{1}{Z} \frac{|Z - Z'|^2}{|\sqrt{Z} + \sqrt{Z'}|^2} \leq K |Z - Z'|^2.$$

Since $\Phi(v;y) \geq 0$ and $\Phi(v;y') \geq 0$, we have from (2.33), (2.34), using the fact that e^{-x} is Lipschitz on \mathbb{R}^+ ,

$$\begin{aligned} |Z - Z'| &\leq Z_0 \int |e^{-\Phi(v;y)} - e^{-\Phi(v;y')}| \rho_0(v) dv \\ &\leq Z_0 \int |\Phi(v;y) - \Phi(v;y')| \rho_0(v) dv. \end{aligned}$$

By definition of Φ and use of (2.37), we have

$$\begin{aligned} |\Phi(v;y) - \Phi(v;y')| &\leq \frac{1}{2} \sum_{j=0}^{J-1} |y_{j+1} - y'_{j+1}|_\Gamma |y_{j+1} + y'_{j+1} - 2h(v_{j+1})|_\Gamma \\ &\leq \frac{1}{2} \left(\sum_{j=0}^{J-1} |y_{j+1} - y'_{j+1}|_\Gamma^2 \right)^{\frac{1}{2}} \left(\sum_{j=0}^{J-1} |y_{j+1} + y'_{j+1} - 2h(v_{j+1})|_\Gamma^2 \right)^{\frac{1}{2}} \\ &\leq K |y - y'| \left(\sum_{j=0}^{J-1} (1 + |h(v_{j+1})|^2) \right)^{\frac{1}{2}} \\ &= K |y - y'| v^{\frac{1}{2}}. \end{aligned}$$

Since $\mathbb{E}^{\mu_0} v < \infty$ implies that $\mathbb{E}^{\mu_0} v^{\frac{1}{2}} < \infty$, it follows that

$$|Z - Z'| \leq K |y - y'|.$$

Hence $I_2 \leq K |y - y'|^2$

Using that Z_0 is bounded above independently of r , that Z is bounded below, depending on data only through r , and that $e^{-\frac{1}{2}x}$ is Lipschitz on \mathbb{R}^+ , it now follows that I_1 satisfies

$$I_1 \leq K \int |\Phi(v;y) - \Phi(v;y')|^2 \rho_0(v) dv.$$

Squaring the preceding bound on $|\Phi(v; y) - \Phi(v; y')|$ gives

$$|\Phi(v; y) - \Phi(v; y')|^2 \leq K|y - y'|^2 v,$$

and so $I_1 \leq K|y - y'|^2$, as required. \square

Corollary 2.16. *Consider the smoothing problem arising from the stochastic dynamics model (2.1), resulting in the posterior probability distributions μ and μ' associated with two different data sets y and y' . Assume that $\mathbb{E}^{\mu_0} v < \infty$, where v is given by (2.36). Let $f : \mathbb{R}^{|\mathbb{J}_0| \times n} \rightarrow \mathbb{R}^p$ be such that $\mathbb{E}^{\mu_0} |f(v)|^2 < \infty$. Then there is $c = c(r) > 0$ such that for all $|y|, |y'| < r$,*

$$|\mathbb{E}^\mu f(v) - \mathbb{E}^{\mu'} f(v)| \leq c|y - y'|.$$

Proof First note that since $\Phi(v; y) \geq 0$, Z_0 is bounded above independently of r , and since Z is bounded from below depending only on r , $\mathbb{E}^\mu |f(v)|^2 \leq c\mathbb{E}^{\mu_0} |f(v)|^2$; and a similar bound holds under μ' . The result follows from (1.13) and Theorem 2.15. \square

Using the relationship between filtering and smoothing as described in the previous section, we may derive a corollary concerning the filtering distribution.

Corollary 2.17. *Consider the smoothing problem arising from the stochastic dynamics model (2.1), resulting in the posterior probability distributions μ and μ' associated with two different data sets y and y' . Assume that $\mathbb{E}^{\mu_0} v < \infty$, where v is given by (2.36). Let $g : \mathbb{R}^n \rightarrow \mathbb{R}^p$ be such that $\mathbb{E}^{\mu_0} |g(v_J)|^2 < \infty$. Then there is $c = c(r) > 0$ such that for all $|y|, |y'| < r$,*

$$|\mathbb{E}^{\mu_J} g(u) - \mathbb{E}^{\mu'_J} g(u)| \leq c|Y_J - Y'_J|,$$

where μ_J and μ'_J denote the filtering distributions at time J corresponding to data Y_J, Y'_J respectively (i.e., the marginals of μ and μ' on the coordinate at time J).

Proof Since by Theorem 2.12, μ_J is the marginal of the smoother on the v_J -coordinate, the result follows from Corollary 2.16 by choosing $f(v) = g(v_J)$. \square

A similar theorem and corollaries may be proved for the case of deterministic dynamics (2.3) and the posterior $\mathbb{P}(v_0|y)$. We state the theorem and leave its proof to the reader. We let ν_0 denote the prior Gaussian measure $N(m_0, C_0)$ on v_0 for the smoothing problem arising in deterministic dynamics, and ν and ν' the posterior measures on v_0 resulting from two different instances of the data, y and y' respectively. We also define

$$v_0 := \sum_{j=0}^{J-1} (1 + |h(\Psi^{(j+1)}(v_0))|^2).$$

Theorem 2.18. *Consider the smoothing problem arising from the deterministic dynamics model (2.3). Assume that $\mathbb{E}^{\nu_0} v_0 < \infty$. Then there is $c = c(r) > 0$ such that for all $|y|, |y'| \leq r$,*

$$d_{\text{Hell}}(\nu, \nu') \leq c|y - y'|.$$

2.7 Assessing the Quality of Data-Assimilation Algorithms

It is helpful in studying algorithms for data assimilation to ask two questions: (i) how informative is the data we have? (ii) how good is our algorithm at extracting the requisite information? These are two separate questions, answers to both of which are required in

the quest to understand how well we perform at extracting a signal, using model and data. We take the two questions separately, in turn; however, we caution that many applied papers entangle them both and simply measure algorithm quality by ability to reconstruct the signal.

Answering question (i) is independent of any particular algorithm: it concerns the properties of the Bayesian posterior pdf itself. In some cases, we will be interested in studying the properties of the probability distribution on the signal, or the initial condition, for a particular instance of the data generated from a particular instance of the signal, which we call the **truth**. In this context, we will use the notation $y^\dagger = \{y_j^\dagger\}$ to denote the *realization* of the data generated from a particular realization of the truth $v^\dagger = \{v_j^\dagger\}$. We first discuss properties of the smoothing problem for stochastic dynamics. **Posterior consistency** concerns the question of the limiting behavior of $\mathbb{P}(v|y^\dagger)$ as either $J \rightarrow \infty$ (large data sets) or $|\Gamma| \rightarrow 0$ (small noise). A key question is whether $\mathbb{P}(v|y^\dagger)$ converges to the truth in either of these limits; this might happen, for example, if $\mathbb{P}(v|y^\dagger)$ becomes closer and closer to a Dirac probability measure centered on v^\dagger . When this occurs, we say that the problem exhibits Bayesian posterior consistency; it is then of interest to study the rate at which the limit is attained. Such questions concern the information content of the data; they do not refer to any algorithm, and therefore, they are not concerned with the quality of any particular algorithm. In considering filtering, rather than smoothing, a particular instance of this question concerns marginal distributions: for example, one may be concerned with posterior consistency of $\mathbb{P}(v_J|y_J^\dagger)$ with respect to a Dirac measure on v_J^\dagger in the filtering case; see Theorem 2.12. For the case of deterministic dynamics, the distribution $\mathbb{P}(v|y^\dagger)$ is completely determined by $\mathbb{P}(v_0|y^\dagger)$ (see Theorem 2.14), so one may discuss posterior consistency of $\mathbb{P}(v_0|y^\dagger)$ with respect to a Dirac measure on v_0^\dagger .

Here it is appropriate to mention the important concept of **model error**. In many (in fact most) applications, the physical system that generates the data set $\{y_j\}$ can be (sometimes significantly) different from the mathematical model used, at least in certain aspects. This can be thought of conceptually by imagining data generated by (2.2), with $v^\dagger = \{v_j^\dagger\}$ governed by the deterministic dynamics

$$v_{j+1}^\dagger = \Psi_{\text{true}}(v_j^\dagger), \quad j \in \mathbb{Z}^+, \quad (2.38a)$$

$$v_0^\dagger = u \sim N(m_0, C_0). \quad (2.38b)$$

Here the function Ψ_{true} governs the dynamics of the truth that underlies the data. We assume that the true solution operator is not known to us exactly, and we seek instead to combine the data with the stochastic dynamics model (2.1); the noise $\{\xi_j\}$ is used to allow for the discrepancy between the true solution operator Ψ_{true} and that used in our model, namely Ψ . It is possible to think of many variants on this situation. For example, the dynamics of the truth may be stochastic; or the dynamics of the truth may take place in a higher-dimensional space than that used in our models, and may need to be projected into the model space. Statisticians sometimes refer to the situation in which the data source differs from the model used as **model misspecification**.

We now turn from the information content, or quality, of the data to the quality of algorithms for data assimilation. We discuss three approaches to assessing quality. The first, fully Bayesian, approach can be defined independently of the quality of the data. The second approach, estimation, entangles the properties of the algorithm with the quality of the data. We discuss these two approaches in the context of the smoothing problem for stochastic dynamics. The reader will easily see how to generalize to smoothing for deterministic dynamics or to filtering. The third approach is widely used in operational numerical weather prediction and judges quality by the ability to predict.

Bayesian Quality Assessment. Here we assume that the algorithm under consideration provides an approximation $\mathbb{P}_{\text{approx}}(v|y)$ to the true posterior distribution $\mathbb{P}(v|y)$. We ask the following question: how close is $\mathbb{P}_{\text{approx}}(v|y)$ to $\mathbb{P}(v|y)$? We might look for a distance measure between probability distributions, or we might simply compare some important moments of the distributions, such as the mean and covariance. Note that this version of quality assessment does not refer to the concept of a true solution v^\dagger . We may apply it with $y = y^\dagger$, but we may also apply it when there is model error present and the data comes from outside the model used to perform data assimilation. However, if combined with Bayesian posterior consistency, when $y = y^\dagger$, then the triangle inequality relates the output of the algorithm to the truth v^\dagger . Very few practitioners evaluate their algorithms by this measure. This reflects the fact that knowing the true distribution $\mathbb{P}(v|y)$ is often difficult in practical high-dimensional problems. However, it is arguably the case that practitioners should spend more time querying their algorithms from the perspective of Bayesian quality assessment, since the algorithms are often used to make probabilistic statements and forecasts.

Signal Estimation Quality Assessment. Here we assume that the algorithm under consideration provides an approximation to the signal v underlying the data, which we denote by v_{approx} ; thus v_{approx} attempts to determine and then *track* the true signal from the data. If the algorithm actually provides a probability distribution, then this estimate might be, for example, the mean. We ask the following question: if the algorithm is applied in the situation in which the data y^\dagger is generated from the signal v^\dagger , how close is v_{approx} to v^\dagger ? There are two important effects at play here: the first is the information content of the data—does the data actually contain enough information to allow for accurate reconstruction of the signal in principle? And the second is the role of the specific algorithm used—does the specific algorithm in question have the ability to extract this information when it is present? This approach thus measures the overall effect of these two in combination.

Forecast Skill. In many cases, the goal of data assimilation is to provide better forecasts of the future, for example in numerical weather prediction. In this context, data-assimilation algorithms can be benchmarked by their ability to make forecasts. This can be discussed in both the Bayesian quality and signal estimation senses. We first discuss Bayesian estimation forecast skill in the context of stochastic dynamics. The Bayesian k -lag forecast skill can be defined by studying the distance between the approximation $\mathbb{P}_{\text{approx}}(v|y)$ and $\mathbb{P}(v|y)$ when both are pushed forward from the endpoint of the data assimilation window by k applications of the dynamical model (2.1); this model defines a Markov transition kernel that is applied k times to produce a forecast. We now discuss signal estimation forecast skill in the context of deterministic dynamics. Using v_{approx} at the endpoint of the assimilation window as an initial condition, we run the model (2.3) forward by k steps and compare the output with v_{j+k}^\dagger . In practical applications, this forecast methodology inherently confronts the effect of model error, since the data used to test forecasts is real data that is not generated by the model used to assimilate, as well as information content in the data and algorithm quality.

2.8 Illustrations

In order to build intuition concerning the probabilistic viewpoint on data assimilation, we describe some simple examples in which the posterior distribution may be visualized easily. For this reason, we concentrate on the case of one-dimensional deterministic dynamics; the posterior pdf $\mathbb{P}(v_0|y)$ for deterministic dynamics is given by Theorem 2.11. It is one-dimensional when the dynamics is one-dimensional and takes place in \mathbb{R} . In Section 3, we will introduce

more sophisticated sampling methods to probe probability distributions in higher dimensions that arise from noisy dynamics and/or from high-dimensional models.

Figure 2.10 concerns the scalar linear problem from Example 2.1 (recall that throughout this section, we consider only the case of deterministic dynamics) with $\lambda = 0.5$. We employ a prior $N(4, 5)$, we assume that $h(v) = v$, and we set $\Gamma = \gamma^2$ and consider two different values of γ and two different values of J , the number of observations. The figure shows the posterior distribution in these various parameter regimes. The true value of the initial condition that underlies the data is $v_0^\dagger = 0.5$. For both $\gamma = 1.0$ and 0.1 , we see that as the number of observations J increases, the posterior distribution appears to converge to a limiting distribution. However, for smaller γ , the limiting distribution has much smaller variance, and is centered closer to the true initial condition at 0.5. Both of these observations can be explained, using the fact that the problem is explicitly solvable: we show that for fixed γ and $J \rightarrow \infty$, the posterior distribution has a limit, which is a Gaussian with nonzero variance. And for fixed J , as $\gamma \rightarrow 0$, the posterior distribution converges to a Dirac measure (Gaussian with zero variance) centered at the truth v_0^\dagger .

To see these facts, we start by noting that from Theorem 2.11, the posterior distribution on $v_0|y$ is proportional to the exponential of

$$\mathcal{I}_{\text{det}}(v_0; y) = \frac{1}{2\gamma^2} \sum_{j=0}^{J-1} |y_{j+1} - \lambda^{j+1} v_0|^2 + \frac{1}{2\sigma_0^2} |v_0 - m_0|^2,$$

where σ_0^2 denotes the prior variance C_0 . As a quadratic form in v_0 , this defines a Gaussian posterior distribution, and we may complete the square to find the posterior mean m and variance σ_{post}^2 :

$$\frac{1}{\sigma_{\text{post}}^2} = \frac{1}{\gamma^2} \sum_{j=0}^{J-1} \lambda^{2(j+1)} + \frac{1}{\sigma_0^2} = \frac{1}{\gamma^2} \left(\frac{\lambda^2 - \lambda^{2J+2}}{1 - \lambda^2} \right) + \frac{1}{\sigma_0^2}$$

and

$$\frac{1}{\sigma_{\text{post}}^2} m = \frac{1}{\gamma^2} \sum_{j=0}^{J-1} \lambda^{(j+1)} y_{j+1} + \frac{1}{\sigma_0^2} m_0.$$

We note immediately that the posterior variance is independent of the data. Furthermore, if we fix γ and let $J \rightarrow \infty$, then for every $|\lambda| < 1$, we see that the large- J limit of the posterior variance is determined by

$$\frac{1}{\sigma_{\text{post}}^2} = \frac{1}{\gamma^2} \left(\frac{\lambda^2}{1 - \lambda^2} \right) + \frac{1}{\sigma_0^2}$$

and is nonzero; thus uncertainty remains in the posterior, even in the limit of large data. On the other hand, if we fix J and let $\gamma \rightarrow 0$, then $\sigma_{\text{post}}^2 \rightarrow 0$, so that uncertainty disappears in this limit. It is then natural to ask what happens to the mean. To this end, we assume that the data is itself generated by the linear model of Example 2.1, so that

$$y_{j+1} = \lambda^{j+1} v_0^\dagger + \gamma \zeta_{j+1},$$

where ζ_j is an i.i.d. Gaussian sequence with $\zeta_1 \sim N(0, 1)$. Then

$$\frac{1}{\sigma_{\text{post}}^2} m = \frac{1}{\gamma^2} \left(\frac{\lambda^2 - \lambda^{2J+2}}{1 - \lambda^2} \right) v_0^\dagger + \frac{1}{\gamma} \sum_{j=0}^{J-1} \lambda^{(j+1)} \zeta_{j+1} + \frac{1}{\sigma_0^2} m_0.$$

Using the formula for σ_{post}^2 , we obtain

$$\left(\frac{\lambda^2 - \lambda^{2J+2}}{1 - \lambda^2}\right)m + \frac{\gamma^2}{\sigma_0^2}m = \left(\frac{\lambda^2 - \lambda^{2J+2}}{1 - \lambda^2}\right)v_0^\dagger + \gamma \sum_{j=0}^{J-1} \lambda^{(j+1)} \zeta_{j+1} + \frac{\gamma^2}{\sigma_0^2}m_0.$$

From this, it follows that for fixed J and as $\gamma \rightarrow 0$, $m \rightarrow v_0^\dagger$, almost surely with respect to the noise realization $\{\zeta_j\}_{j \in \mathbb{J}}$. This is an example of posterior consistency.

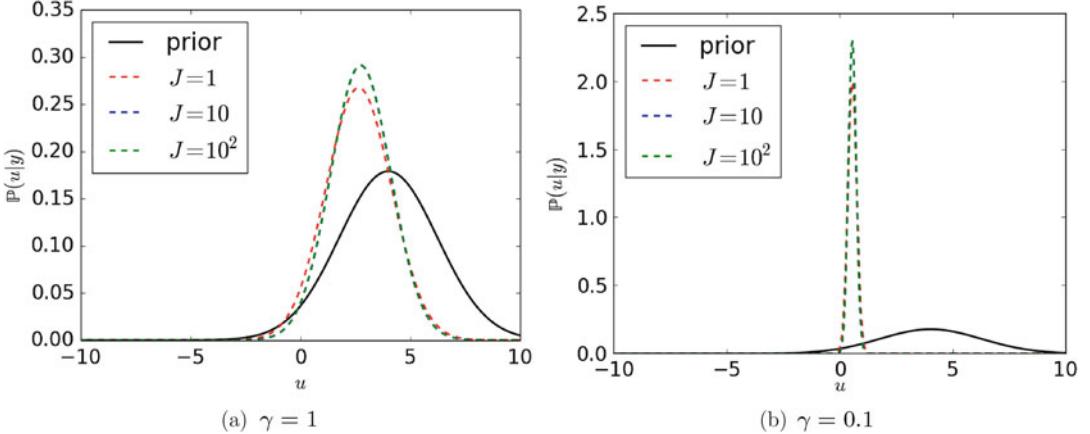


Fig. 2.10: Posterior distribution for Examples 2.1 for different levels of observational noise. The true initial condition used in both cases is $v_0 = 0.5$, while we have assumed that $C_0 = 5$ and $m_0 = 4$ for the prior distribution.

We now study Example 2.4, in which the true dynamics are no longer linear. We begin our investigation taking $r = 2$, and we investigate the effect of choosing different prior distributions. Before discussing the properties of the posterior, we draw attention to two facts. Firstly, as Figure 2.5a shows, the system converges in a small number of steps to the fixed point at $1/2$ for this value of $r = 2$. And secondly, the initial conditions v_0 and $1 - v_0$ both result in the same trajectory if the initial condition is ignored. The first point implies that after a small number of steps, the observed trajectory contains very little information about the initial condition. The second point means that since we observe from the first step onward, only the prior can distinguish between v_0 and $1 - v_0$ as the starting point.

Figure 2.11 concerns an experiment in which the true initial condition underlying the data is $v_0^\dagger = 0.1$. Two different priors are used, both with $C_0 = 0.01$, giving a standard deviation of 0.1, but with different means. The figure illustrates two facts: firstly, even with 10^3 observations, the posterior contains considerable uncertainty, reflecting the first point above. Secondly, the prior mean has an important role in the form of the posterior pdf: shifting the prior mean to the right, from $m_0 = 0.4$ to $m_0 = 0.7$, results in a posterior that favors the initial condition $1 - v_0^\dagger$ rather than the truth v_0^\dagger .

This behavior of the posterior changes completely if we assume a flatter prior. This is illustrated in Figure 2.12, where we consider the prior $N(0.4, C_0)$ with $C_0 = 0.5$ and 5 respectively. As we increase the prior covariance, the mean plays a much weaker role than in the preceding experiments: we now obtain a bimodal posterior centered at both the true initial condition v_0^\dagger , and also at $1 - v_0^\dagger$.

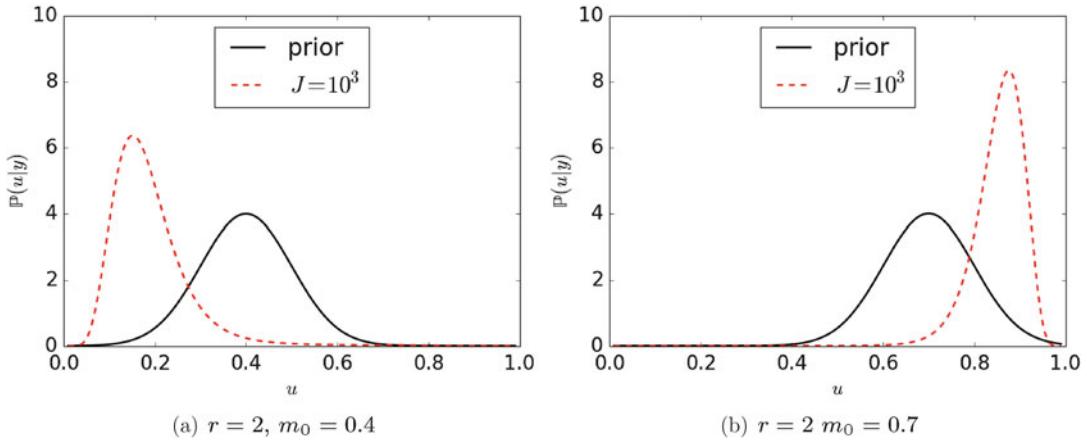


Fig. 2.11: Posterior distribution for Example 2.4 for $r = 2$ in the case of different means for the prior distribution. We have used $C_0 = 0.01$, $\gamma = 0.1$, and true initial condition $v_0 = 0.1$; see also `p2.m` in Section 5.1.2.

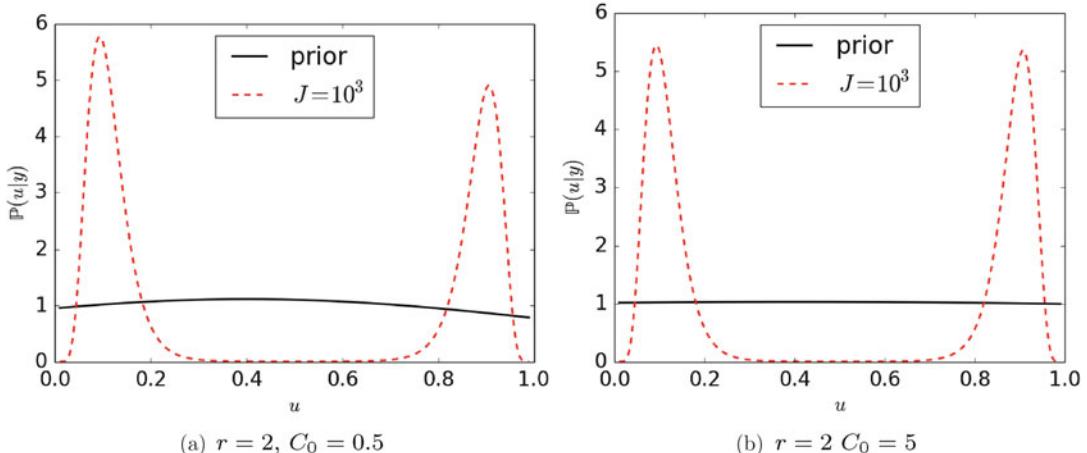


Fig. 2.12: Posterior distribution for Example 2.4 for $r = 2$ in the case of different covariance for the prior distribution. We have used $m_0 = 0.4$, $\gamma = 0.1$, and true initial condition $v_0 = 0.1$.

In Figure 2.13, we consider the quadratic map (2.9) with $r = 4$, $J = 5$, and prior $N(0.5, 0.01)$, with observational standard deviation $\gamma = 0.2$. Here, after only five observations, the posterior is very peaked, although because of the $v \mapsto 1 - v$ symmetry mentioned above, there are two symmetrically related peaks; see Figure 2.13a. It is instructive to look at the negative of the logarithm of the posterior pdf, which, up to an additive constant, is given by $I_{\text{det}}(v_0; y)$ in Theorem 2.11. The function $I_{\text{det}}(\cdot; y)$ is shown in Figure 2.13b. Its complexity indicates the considerable complications underlying the solution of the smoothing problem. We will return to this last point in detail later. Here we simply observe that normalizing the posterior distribution requires evaluation of the integral

$$\int_{\mathbb{P}^n} e^{-l(v_0, y)} dv_0.$$

This integral may often be determined almost entirely by very small subsets of \mathbb{R}^n , meaning that this calculation requires some care; indeed, if $I(\cdot)$ is very large over much of its domain, then it may be impossible to compute the normalization constant numerically. We note, however, that the sampling methods that we will describe in the next chapter do not require evaluation of this integral.

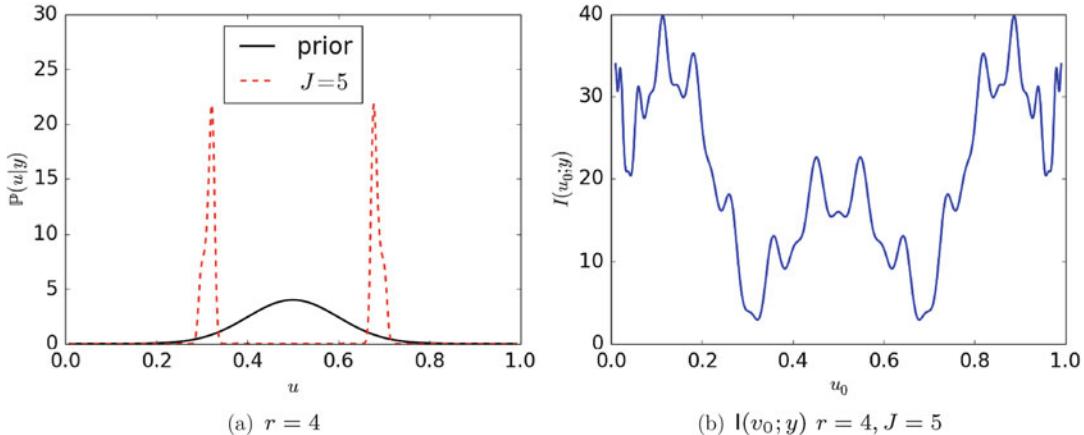


Fig. 2.13: Posterior distribution and negative log-posterior for Example 2.4 for $r = 4$ and $J = 5$. We have used $C_0 = 0.01$, $m_0 = 0.5$, $\gamma = 0.2$, and true initial condition $v_0 = 0.3$.

2.9 Bibliographic Notes

- Section 2.1, Data Assimilation, has its roots in the geophysical sciences and is driven by the desire to improve inaccurate models of complex dynamically evolving phenomena by means of incorporation of data. The book [81] describes data assimilation from the viewpoint of the atmospheric sciences and weather prediction, while the book [12] describes the subject from the viewpoint of oceanography. These two subjects were the initial drivers for evolution of the field. However, other applications are increasingly using the methodology of data assimilation, and the oil industry in particular is heavily involved in the use, and development, of algorithms in this area [115]. The recent book [1] provides a perspective on the subject from the viewpoint of physics and nonlinear dynamical systems, and includes motivational examples from neuroscience, as well as the geophysical sciences. The article [74] is a useful one to read because it establishes a notation that is now widely used in the applied communities, and the articles [111, 6] provide simple introductions to various aspects of the subject from a mathematical perspective. The special edition of the journal *Physica D* devoted to data assimilation, [75], provides an overview of the state of the art around a decade ago.
- It is useful to comment on generalizations of the setup described in Section 2.1. First we note that we have assumed a *Gaussian* structure for the additive noise appearing in both the signal model (2.1) and the data model (2.2). This is easily relaxed in much of what we describe here, provided that an explicit formula for the probability density function of the noise is known. However, the Kalman filter, described in the next chapter, relies

explicitly on the closed Gaussian form of the probability distributions resulting from the assumption of Gaussian noise. There are also other parts of the notes, such as the pCN MCMC methods and the minimization principle underlying approximate Gaussian filters, also both described in the next chapter, that require the Gaussian structure. Second, we note that we have assumed *additive* noise. This, too, can be relaxed, but that has the complication that most nonadditive noise models do not yield explicit formulas for the needed conditional probability density functions; for example, this situation arises if one looks at stochastic differential equations over discrete time intervals—see [17] and the discussion therein. However, some of the methods we describe rely only on drawing samples from the desired distributions and do not require the explicit conditional probability density function. Finally, we note that much of what we describe here translates to *infinite-dimensional* spaces with respect to both the signal space and the data space; however, in the case of an infinite-dimensional data space, the additive Gaussian observational noise is currently the only situation that is well developed [133].

- Section 2.2. The subject of deterministic discrete-time dynamical systems of the form (2.3) is reviewed in numerous texts; see [147] and the references therein, and Chapter 1 of [134], for example. The subject of stochastic discrete-time dynamical systems of the form (2.1), and in particular the property of ergodicity that underlies Figure 2.4, is covered in some depth in [108]. The exact solutions of the quadratic map (2.9) for $r = 2$ and $r = 4$ may be found in [127] and [96] respectively. The Lorenz '63 model was introduced in [95]. Not only does this paper demonstrate the possibility of chaotic behavior and sensitivity with respect to initial conditions, but it also makes a concrete connection between a three-dimensional continuous-time dynamical system and a one-dimensional chaotic map of the form (2.1). Furthermore, a subsequent computer-assisted proof demonstrated rigorously that the ODE indeed exhibits chaos [139, 140]. The book [132] discusses properties of the Lorenz '63 model in some detail, and the book [51] discusses properties such as fractal dimension. The shift of origin that we have adopted for the Lorenz '63 model is explained in [137]; it enables the model to be written in an abstract form that includes many geophysical models of interest, such as the Lorenz '96 model, introduced in [97], and the Navier–Stokes equation on a two-dimensional torus [102, 137]. We now briefly describe this common abstract form. The vector $u \in \mathbb{R}^J$ ($J = 3$ for Lorenz '63, J arbitrary for Lorenz' 96) solves the equation

$$\frac{du}{dt} + Au + B(u, u) = f, \quad u(0) = u_0, \quad (2.39)$$

where there is $\lambda > 0$ such that for all $w \in \mathbb{R}^J$,

$$\langle Aw, w \rangle \geq \lambda|w|^2, \quad \langle B(w, w), w \rangle = 0.$$

Taking the inner product with u shows that

$$\frac{1}{2} \frac{d}{dt} |u|^2 + \lambda|u|^2 \leq \langle f, u \rangle.$$

If f is constant in time, then this inequality may be used to show that (2.16) holds:

$$\frac{1}{2} \frac{d}{dt} |u|^2 \leq \frac{1}{2\lambda} |f|^2 - \frac{\lambda}{2} |u|^2.$$

Integrating this inequality gives the existence of an absorbing set and hence leads to the existence of a global attractor; see Example 1.22, the book [137], or Chapter 2 of [134], for example.

- Section 2.3 contains the formulation of data assimilation as a fully nonlinear and non-Gaussian problem in Bayesian statistics. This formulation is not yet the basis of practical algorithms in geophysical systems such as weather forecasting. This is because global weather forecast models involve $n = \mathcal{O}(10^9)$ unknowns, and incorporate $m = \mathcal{O}(10^6)$ data points daily; sampling the posterior on \mathbb{R}^n given data in \mathbb{R}^m in an online fashion, usable for forecasting, is beyond current algorithmic and computational capability. However, the fully Bayesian perspective provides a fundamental mathematical underpinning of the subject, from which other more tractable approaches can be systematically derived. See [133] for a discussion of the Bayesian approach to inverse problems. Historically, data assimilation did not evolve from this Bayesian perspective, but rather evolved out of the control theory perspective. This perspective is summarized well in the book [77]. However, the importance of the Bayesian perspective is increasingly being recognized in the applied communities. In addition to providing a starting point from which to derive approximate algorithms, it also provides a gold standard against which other more ad hoc algorithms can be benchmarked; this use of Bayesian methodology was suggested in [89] in the context of meteorology (see discussion that follows), and then employed in [76] in the context of subsurface inverse problems arising in geophysics.
- Section 2.4 describes the filtering, or sequential, approach to data assimilation, within the fully Bayesian framework. For low-dimensional systems, the use of particle filters, which may be shown to approximate the required filtering distribution rigorously as it evolves in discrete time, has been enormously successful; see [48] for an overview. Unfortunately, these filters can behave poorly in high dimensions [19, 11, 129]. While there is ongoing work to overcome these problems with high-dimensional particle filtering, see [16, 28, 143], for example, this work has yet to impact practical data assimilation in, for example, operational weather forecasting. For this reason, the ad hoc filters, such as the 3DVAR filter, extended Kalman filter, and ensemble Kalman filter, described in Chapter 4, are of great practical importance. Their analysis is hence an important challenge for applied mathematicians.
- Section 2.6, on data assimilation, may be viewed as an inverse problem to determine the signal from the observations. Inverse problems in differential equations are often ill posed when viewed from a classical nonprobabilistic perspective. One reason for this is that the data may not be informative about the whole signal, so that many solutions are possible. However, taking the Bayesian viewpoint, in which the many solutions are all given a probability, allows for well-posedness to be established. This idea is used for data-assimilation problems arising in fluid mechanics in [32], for inverse problems arising in subsurface geophysics in [44, 42], and described more generally in [133]. Well-posedness with respect to changes in the data is of importance in its own right, but also more generally because it underpins other stability results that can be used to control perturbations. In particular, the effect of numerical approximation on integration of the forward model can be understood in terms of its effect on the posterior distribution; see [33]. A useful overview of probability metrics, including Hellinger and total-variation metrics, is contained in [57].
- Section 2.7. The subject of posterior consistency is central to the theory of statistics in general [141], and within Bayesian statistics in particular [13, 15, 56]. Assessing the quality of data assimilation algorithms is typically performed in the “signal estimation” framework using **identical twin experiments** in which the data is generated from the same model used to estimate the signal; see [75] and the references therein. The idea of assessing “Bayesian quality” has only recently been used within the data-assimilation literature; see [89], where this approach is taken for the Navier–Stokes inverse problem formulated in [32]. The evaluation of algorithms by means of forecast skill is enormously influential in the field of numerical weather prediction and drives a great deal of algorithmic selection. The use of information theory to understand the effects of model error and to evaluate

filter performance is introduced in [99] and [22] respectively. There are also a number of useful **consistency checks** that can be applied to evaluate the computational model and its fit to the data [50, 2, 4]. We discuss the idea of the variant known as *rank histograms* at the end of Chapter 4. If the empirical statistics of the innovations are inconsistent with the assumed model, then they can be used to improve the model in the future; this is known as *reanalysis*.

2.10 Exercises

1. Consider the map given in Example 2.3 and related program p1.m. By experimenting with the code, determine approximately the value of α , denoted by α_1 , at which the noise-free dynamics changes from convergence to an equilibrium point to convergence to a period-2 solution. Can you find a value of $\alpha = \alpha_2$ for which you obtain a period-4 solution? Can you find a value of $\alpha = \alpha_3$ for which you obtain a nonrepeating (chaotic) solution? For the values $\alpha = 2, \alpha_2$, and α_3 , compare the trajectories of the dynamical system obtained with the initial condition 1 and with the initial condition 1.1. Comment on what you find. Now fix the initial condition at 1 and consider the same values of α , with and without noise ($\sigma \in \{0, 1\}$). Comment on what you find. Illustrate your answers with graphics. To get interesting displays, you will find it helpful to change the value of J (number of iterations) depending upon what you are illustrating.
2. Consider the map given in Example 2.4 and verify the explicit solutions given for $r = 2$ and $r = 4$ in formulas (2.10)–(2.12).
3. Consider the Lorenz '63 model given in Example 2.6. Determine values of $\{\alpha, \beta\}$ for which (2.16) holds.
4. Consider the Lorenz '96 model given in Example 2.7. Program p19.m plots solutions of the model, as well as analyzing sensitivity to initial conditions. Study the behavior of the equation for $J = 40, F = 2$, for $J = 40, F = 4$, and report your results. Fix F at 8 and play with the value of the dimension of the system, J . Report your results. Again, illustrate your answers with graphics.
5. Consider the posterior smoothing distribution from Theorem 2.8. Assume that the stochastic dynamics model (2.1) is scalar and defined by $\Psi(v) = av$ for some $a \in \mathbb{R}$ and $\Sigma = \sigma^2$, and that the observation model (2.2) is defined by $h(v) = v$ and $\Gamma = \gamma^2$. Find explicit formulas for $J(v)$ and $\Phi(v; y)$, assuming that $v_0 \sim N(m_0, \sigma_0^2)$.
6. Consider the posterior smoothing distribution from Theorem 2.11. Assume that the dynamics model (2.3a) is scalar and defined by $\Psi(v) = av$ for some $a \in \mathbb{R}$, and that the observation model (2.2) is defined by $h(v) = v$ and $\Gamma = \gamma^2$. Find explicit formulas for $J_{\text{det}}(v_0)$ and $\Phi_{\text{det}}(v_0; y)$, assuming that $v_0 \sim N(m_0, \sigma_0^2)$.
7. Consider the definition of total variation distance given in Definition 1.27. State and prove a theorem analogous to Theorem 2.15, but employing the total variation distance instead of the Hellinger distance.
8. Consider the filtering distribution from Section 2.4 in the case that the stochastic dynamics model (2.1) is scalar and defined by $\Psi(v) = av$ for some $a \in \mathbb{R}$ and $\Sigma = \sigma^2$, and that the observation model (2.2) is defined by $h(v) = v$ and $\Gamma = \gamma^2$, and $v_0 \sim N(m_0, \sigma_0^2)$. Demonstrate that the prediction and analysis steps preserve Gaussianity, so that $\mu_j = N(m_j, \sigma_j^2)$. Find iterative formulas that update (m_j, σ_j^2) to give $(m_{j+1}, \sigma_{j+1}^2)$.
9. Prove Theorem 2.18.

Chapter 3

Discrete Time: Smoothing Algorithms

The formulation of the data-assimilation problem described in the previous chapter is probabilistic, and its computational resolution requires the probing of a posterior probability distribution on signal-given data. This probability distribution is on the signal sequence $v = \{v_j\}_{j=0}^J$ when the underlying dynamics is stochastic and given by (2.1); the posterior is specified in Theorem 2.8 and is proportional to $\exp(-l(v; y))$, given by (2.21). On the other hand, if the underlying dynamics is deterministic and given by (2.3), then the probability distribution is on the initial condition v_0 only; it is given in Theorem 2.11 and is proportional to $\exp(-l_{\text{det}}(v_0; y))$, with l_{det} given by (2.29). Generically, in this chapter, we refer to the unknown variable as u , and then use v in the specific case of stochastic dynamics and v_0 in the specific case of deterministic dynamics. The aim of this chapter is to understand $\mathbb{P}(u|y)$. In this regard, we will do three things:

- find explicit expressions for the pdf $\mathbb{P}(u|y)$ in the linear, Gaussian setting;
- generate samples $\{u^{(n)}\}_{n=1}^N$ from $\mathbb{P}(u|y)$ by algorithms applicable in the non-Gaussian setting;
- find points where $\mathbb{P}(u|y)$ is maximized with respect to u , for given data y .

In general, the probability distributions of interest cannot be described by a finite set of parameters, except in a few simple situations such as the Gaussian scenario, in which the mean and covariance determine the distribution in its entirety—the **Kalman smoother**. When the probability distributions cannot be described by a finite set of parameters, an expedient computational approach to representing the measure approximately is through the idea of **Monte Carlo sampling**. The basic idea is to approximate a measure ν by a set of N samples $\{u^{(n)}\}_{n \in \mathbb{Z}^+}$ drawn, or approximately drawn, from ν to obtain the measure $\nu^N \approx \nu$ given by

$$\nu^N = \frac{1}{N} \sum_{n=1}^N \delta_{u^{(n)}}. \quad (3.1)$$

We may view this as defining a (random) map S^N on measures that takes ν into ν^N . If the $u^{(n)}$ are exact draws from ν , then the resulting approximation ν^N converges to the true measure ν as $N \rightarrow \infty$.¹ For example, if $v = \{v_j\}_{j=0}^J$ is governed by the probability distribution μ_0 defined by the unconditioned dynamics (2.1), and with pdf determined by (2.19), then exact

Electronic supplementary material The online version of this chapter (doi: [10.1007/978-3-319-20325-6_3](https://doi.org/10.1007/978-3-319-20325-6_3)) contains supplementary material, which is available to authorized users.

¹ Indeed, we prove such a result in Lemma 4.7 in the context of the particle filter.

independent samples $v^{(n)} = \{v_j^{(n)}\}_{j=0}^J$ are easy to generate, simply by running the dynamics model forward in discrete time. However, for the complex probability measures of interest here, where the signal is conditioned on data, exact samples are typically not possible, and so instead, we use the idea of **Monte Carlo Markov chain (MCMC)** methods, which provide a methodology for generating approximate samples. These methods do not require knowledge of the normalization constant for the measure $\mathbb{P}(u|y)$; as we have discussed, Bayes's formula (1.7) readily delivers $\mathbb{P}(u|y)$ up to normalization, but the normalization itself can be difficult to compute. It is also of interest simply to maximize the posterior probability distribution to find a single-point estimate of the solution, leading to **variational methods**, which we also consider.

Section 3.1 gives explicit formulas for the solution of the smoothing problem in the setting in which the stochastic dynamics model is linear, and subject to Gaussian noise, for which the observation operator is linear, and for which the distributions of the initial condition and the observational noise are Gaussian; this is the Kalman smoother. These explicit formulas help to build intuition about the nature of the smoothing distribution. In Section 3.2, we provide some background concerning MCMC methods, and in particular, the **Metropolis–Hastings** variant of MCMC, and show how they can be used to explore the posterior distribution. It can be very difficult to sample the probability distributions of interest with high accuracy, because of the two problems of high dimension and sensitive dependence on initial conditions. While we do not claim to introduce the optimal algorithms to deal with these issues, we do discuss such issues in relation to the samplers we introduce, and we provide references to the active research ongoing in this area. Furthermore, although sampling of the posterior distribution may be computationally infeasible in many situations, it provides, where possible, an important benchmark solution, enabling other algorithms to be compared against a “gold standard” Bayesian solution.

However, because sampling the posterior distribution can be prohibitively expensive, a widely used computational methodology is simply to find the point that maximizes the probability, using techniques from optimization. These are the variational methods, also known as **4DVAR** and **weak constraint 4DVAR**. We introduce this approach to the problem in Section 3.3. In Section 3.4, we provide numerical illustrations that showcase the MCMC and variational methods. The chapter concludes with bibliographic notes in Section 3.5 and exercises in Section 3.6.

3.1 Linear Gaussian Problems: The Kalman Smoother

The Kalman smoother plays an important role, because it is one of the few examples for which the smoothing distribution can be explicitly determined. This explicit characterization occurs because the signal dynamics and observation operator are assumed to be linear. When combined with the Gaussian assumptions on the initial condition for the signal, and on the signal and observational noise, this gives rise to a posterior smoothing distribution that is also Gaussian.

To find formulas for this Gaussian Kalman smoothing distribution, we set

$$\Psi(v) = Mv, \quad h(v) = Hv \tag{3.2}$$

for matrices $M \in \mathbb{R}^{n \times n}$, $H \in \mathbb{R}^{m \times n}$ and consider the signal/observation model (2.1), (2.2). Given data $y = \{y_j\}_{j \in \mathbb{J}}$ and signal $v = \{v_j\}_{j \in \mathbb{J}_0}$, we are interested in the probability distribution of $v|y$, as characterized in Section 2.3.2. By specifying the linear model (3.2) and applying Theorem 2.8, we obtain the following result:

Theorem 3.1. *The posterior smoothing distribution on $v|y$ for the linear stochastic dynamics model (2.1), (2.2), (3.2) with C_0 , Σ and Γ symmetric positive definite is a Gaussian probability measure $\mu = N(m, C)$ on $\mathbb{R}^{|\mathbb{J}_0| \times n}$. The covariance C is the inverse of a symmetric positive definite block tridiagonal precision matrix*

$$L = \begin{pmatrix} L_{11} & L_{12} & & & \\ L_{21} & L_{22} & L_{23} & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \ddots & L_{JJ+1} \\ & & & L_{J+1J} & L_{J+1J+1} \end{pmatrix}$$

with $L_{ij} \in \mathbb{R}^{n \times n}$ given by $L_{11} = C_0^{-1} + M^T \Sigma^{-1} M$, $L_{jj} = H^T \Gamma^{-1} H + M^T \Sigma^{-1} M + \Sigma^{-1}$ for $j = 2, \dots, J$, $L_{J+1,J+1} = H^T \Gamma^{-1} H + \Sigma^{-1}$, $L_{jj+1} = -M^T \Sigma^{-1}$, and $L_{j+1j} = -\Sigma^{-1} M$ for $j = 1, \dots, J$. Furthermore, the mean m solves the equation

$$Lm = r,$$

where

$$r_1 = C_0^{-1} m_0, \quad r_j = H^T \Gamma^{-1} y_{j-1}, \quad j = 2, \dots, J+1.$$

This mean is also the unique minimizer of the functional

$$\begin{aligned} I(v; y) &= \frac{1}{2} |C_0^{-1/2}(v_0 - m_0)|^2 + \sum_{j=0}^{J-1} \frac{1}{2} |\Sigma^{-1/2}(v_{j+1} - Mv_j)|^2 + \sum_{j=0}^{J-1} \frac{1}{2} |\Gamma^{-1/2}(y_{j+1} - Hv_{j+1})|^2 \\ &= \frac{1}{2} |v_0 - m_0|_{C_0}^2 + \sum_{j=0}^{J-1} \frac{1}{2} |v_{j+1} - Mv_j|_\Sigma^2 + \sum_{j=0}^{J-1} \frac{1}{2} |y_{j+1} - Hv_{j+1}|_\Gamma^2 \end{aligned} \quad (3.3)$$

with respect to v and as such is a maximizer, with respect to v , for the posterior pdf $\mathbb{P}(v|y)$.

Proof The proof is based on Lemma 1.6 and the identification of the mean and covariance by studying an appropriate quadratic form. From Theorem 2.8, we know that the desired distribution has pdf proportional to $\exp(-I(v; y))$, where $I(v; y)$ is given in (3.3). This is a quadratic form in v , and we deduce that the inverse covariance L is given by $\partial_v^2 I(v; y)$, the Hessian of I with respect to v . To determine L , we note the following identities:

$$\begin{aligned} D_{v_0}^2 I(v; y) &= C_0^{-1} + M^T \Sigma^{-1} M, \\ D_{v_j}^2 I(v; y) &= \Sigma^{-1} + M^T \Sigma^{-1} M + H^T \Gamma^{-1} H, \quad j = 1, \dots, J-1, \\ D_{v_J}^2 I(v; y) &= \Sigma^{-1} + H^T \Gamma^{-1} H, \\ D_{v_j, v_{j+1}}^2 I(v; y) &= -M^T \Sigma^{-1}, \\ D_{v_{j+1}, v_j}^2 I(v; y) &= -\Sigma^{-1} M. \end{aligned}$$

We may then complete the square and write

$$I(v; y) = \frac{1}{2} \langle (v - m), L(v - m) \rangle + q,$$

where q is independent of v . From this, it follows that the mean indeed minimizes $I(v; y)$ with respect to v , and hence maximizes $\mathbb{P}(v|y) \propto \exp(-I(v; y))$ with respect to v . By differentiating with respect to v , we obtain

$$Lm = r, \quad r = -\nabla_v I(v; y) \Big|_{v=0}, \quad (3.4)$$

where ∇_v is the gradient of I with respect to v . This characterization of r gives the desired equation for the mean. Finally, we show that L , and hence C , is positive definite symmetric. Clearly, L is symmetric, and hence so is C . It remains to check that L is strictly positive definite. To see this, note that if we set $m_0 = 0$, then

$$\frac{1}{2} \langle v, Lv \rangle = I(v; 0) \geq 0. \quad (3.5)$$

Moreover, $I(v; 0) = 0$ with $m_0 = 0$ implies, since $C_0 > 0$ and $\Sigma > 0$,

$$\begin{aligned} v_0 &= 0, \\ v_{j+1} &= Mv_j, \quad j = 0, \dots, J-1, \end{aligned}$$

i.e., $v = 0$. Hence we have shown that $\langle v, Lv \rangle = 0$ implies $v = 0$, and the proof is complete. \square

We now consider the Kalman smoother in the case of deterministic dynamics. Application of Theorem 2.11 gives the following:

Theorem 3.2. *The posterior smoothing distribution on $v_0|y$ for the deterministic linear dynamics model (2.3), (2.2), (3.2) with C_0 and Γ symmetric positive definite is a Gaussian probability measure $\nu = N(m_{\text{det}}, C_{\text{det}})$ on \mathbb{R}^n . The covariance C_{det} is the inverse of the positive definite symmetric matrix L_{det} given by the expression*

$$L_{\text{det}} = C_0^{-1} + \sum_{j=0}^{J-1} (M^T)^{j+1} H^T \Gamma^{-1} H M^{j+1}.$$

The mean m_{det} solves

$$L_{\text{det}} m_{\text{det}} = C_0^{-1} m_0 + \sum_{j=0}^{J-1} (M^T)^{j+1} H^T \Gamma^{-1} y_{j+1}.$$

This mean is a minimizer of the functional

$$I_{\text{det}}(v_0; y) = \frac{1}{2} |v_0 - m_0|_{C_0}^2 + \sum_{j=0}^{J-1} \frac{1}{2} |y_{j+1} - H M^{j+1} v_0|_{\Gamma}^2 \quad (3.7)$$

with respect to v_0 and as such is a maximizer, with respect to v_0 , of the posterior pdf $\mathbb{P}(v_0|y)$.

Proof By Theorem 2.11, we know that the desired distribution has pdf proportional to $\exp(-I_{\text{det}}(v_0; y))$ given by (3.7). The inverse covariance L_{det} can be found as the Hessian of I_{det} , $L_{\text{det}} = \partial_v^2 I_{\text{det}}(v_0; y)$, and the mean m_{det} solves

$$L_{\text{det}} m_{\text{det}} = -\nabla_v I_{\text{det}}(v_0; y) \Big|_{v_0=0}. \quad (3.8)$$

As in the proof of the preceding theorem, we have that

$$I_{\text{det}}(v_0; y) = \frac{1}{2} \langle L_{\text{det}}(v_0 - m_{\text{det}}), (v_0 - m_{\text{det}}) \rangle + q,$$

where q is independent of v_0 ; this shows that m_{det} minimizes $I_{\text{det}}(\cdot; y)$ and maximizes $\mathbb{P}(\cdot|y)$.

We have thus characterized L_{\det} and m_{\det} , and using this characterization gives the desired expressions. It remains to check that L_{\det} is positive definite, since it is clearly symmetric by definition. Positive-definiteness follows from the assumed positive-definiteness of C_0 and Γ , since for every nonzero $v_0 \in \mathbb{R}^n$,

$$\langle v_0, L_{\det} v_0 \rangle \geq \langle v_0 C_0^{-1} v_0 \rangle > 0. \quad (3.9)$$

□

3.2 Markov Chain–Monte Carlo Methods

In the case of stochastic dynamics, equation (2.1), the posterior distribution of interest is the measure μ on $\mathbb{R}^{|\mathbb{J}_0| \times n}$, with density $\mathbb{P}(v|y)$ given in Theorem 2.8; in the case of deterministic dynamics, equation (2.3), it is the measure ν on \mathbb{R}^n with density $\mathbb{P}(v_0|y)$ given in Theorem 2.11. In this section, we describe the idea of Markov chain–Monte Carlo (MCMC) methods for exploring such probability distributions.

We will begin by describing the general MCMC methodology, after which we discuss the specific Metropolis–Hastings instance of this methodology. This material makes no reference to the specific structure of our sampling problem; it works in the general setting of creating a Markov chain that is invariant for an arbitrary measure μ on \mathbb{R}^ℓ with pdf ρ . We then describe applications of the Metropolis–Hastings family of MCMC methods to the smoothing problems of noise-free dynamics and noisy dynamics respectively. In describing the generic Metropolis–Hastings methodology, we will use u (with indices) to denote the state of the Markov chain, and w (with indices) to denote the proposed moves. Thus in the case of stochastic dynamics, the current state u and proposed state w live in the space where signal sequences v lie, and in the case of deterministic dynamics, they live in the space where the initial conditions v_0 lie.

3.2.1. The MCMC Methodology

Recall the concept of a Markov chain $\{u^{(n)}\}_{n \in \mathbb{Z}^+}$ introduced in Section 1.4.1. The idea of MCMC methods is to construct a Markov chain that is invariant with respect to a given measure μ on \mathbb{R}^ℓ and, of particular interest to us, a measure μ with positive Lebesgue density ρ on \mathbb{R}^ℓ . We now use a superscript n to denote the index of the resulting Markov chain, instead of a subscript j , to provide a clear distinction between the Markov chain defined by the stochastic (respectively deterministic) dynamics model (2.1) (respectively (2.3)) and the Markov chains that we will use to sample the posterior distribution on the signal v (respectively initial condition v_0) given data y .

We have already seen that Markov chains allow the computation of averages with respect to the invariant measure by computing the running time-average of the chain—see (1.16). More precisely, we have the following theorem (for which it is useful to recall the notation for the iterated kernel p^n from the very end of Section 1.4.1):

Theorem 3.3. *Assume that if $u^{(0)} \sim \mu$ with Lebesgue density ρ , then $u^{(n)} \sim \mu$ for all $n \in \mathbb{Z}^+$, so that μ is invariant for the Markov chain. If in addition, the Markov chain is ergodic, then for every bounded continuous $\varphi : \mathbb{R}^\ell \rightarrow \mathbb{R}$,*

$$\frac{1}{N} \sum_{n=1}^N \varphi(u^{(n)}) \xrightarrow{a.s.} \mathbb{E}^\mu \varphi(u)$$

for μ a.e. initial condition $u^{(0)}$. In particular, if there exist a probability measure \mathbf{p} on \mathbb{R}^ℓ and $\varepsilon > 0$ such that for all $u \in \mathbb{R}^\ell$ and all Borel sets $A \subseteq \mathcal{B}(\mathbb{R}^\ell)$, we have $p(u, A) \geq \varepsilon \mathbf{p}(A)$, then for all $u \in \mathbb{R}^\ell$,

$$d_{\text{TV}}(p^n(u, \cdot), \mu) \leq 2(1 - \varepsilon)^n. \quad (3.10)$$

Furthermore, there is then $K = K(\varphi) > 0$ such that

$$\frac{1}{N} \sum_{n=1}^N \varphi(u^{(n)}) = \mathbb{E}^\mu \varphi(u) + K \xi_N N^{-\frac{1}{2}}, \quad (3.11)$$

where ξ_N converges weakly to $N(0, 1)$ as $N \rightarrow \infty$.

Remark 3.4. This theorem is the backbone of MCMC. As we will see, there is a large class of methods that ensure invariance of a given measure μ , and furthermore, these methods are often provably ergodic, so that the preceding theorem applies. As with all algorithms in computational science, the optimal algorithm is the one that delivers the smallest error for given unit computational cost. In this regard, there are two observations to make about the preceding theorem.

- The constant K measures the size of the variance of the estimator of $\mathbb{E}^\mu \varphi(x)$, multiplied by N . It is thus a surrogate for the error incurred in running MCMC over a finite number of steps. The constant K depends on φ itself, but it will also reflect general properties of the Markov chain. For a given MCMC method, there will often be tunable parameters whose choice will affect the size of K without affecting the cost per step of the Markov chain. The objective of choosing these parameters is to minimize the constant K , within a given class of methods all of which have the same cost per step. In thinking about how to do this, it is important to appreciate that K measures the amount of correlation in the Markov chain; lower correlation leads to a decreased constant K . More precisely, K is computed by integrating the autocorrelation of the Markov chain.
- A further tension in designing MCMC methods is in the choice of the class of methods themselves. Some Markov chains are expensive to implement, but the convergence in (3.11) is rapid (the constant K can be made small by appropriate choice of parameters), while other Markov chains are cheaper to implement, but the convergence in (3.11) is slower (the constant K is much larger). Some compromise between ease of implementation and rate of convergence needs to be made.



3.2.2. Metropolis–Hastings Methods

The idea of Metropolis–Hastings methods is to build an MCMC method for a measure μ by adding an accept/reject test on top of a Markov chain that is easy to implement but that is not invariant with respect to μ ; the accept/reject step is designed to enforce invariance with respect to μ . This is done by enforcing *detailed balance*:

$$\rho(u)p(u, w) = \rho(w)p(w, u) \quad \forall u, w \in \mathbb{R}^\ell \times \mathbb{R}^\ell. \quad (3.12)$$

Note that integrating with respect to u and using the fact that

$$\int_{\mathbb{R}^\ell} p(w, u) du = 1,$$

we obtain

$$\int_{\mathbb{R}^\ell} \rho(u) p(u, w) du = \rho(w),$$

so that (1.18) is satisfied, and the density ρ is indeed invariant. We now exhibit an algorithm designed to satisfy detailed balance by correcting a given Markov chain, which is not invariant with respect to μ , by the addition of an accept/reject mechanism.

We are given a probability density function ρ , hence satisfying $\rho : \mathbb{R}^\ell \rightarrow \mathbb{R}^+$, with $\int \rho(u) du = 1$. Now consider a Markov transition kernel $q : \mathbb{R}^\ell \times \mathbb{R}^\ell \rightarrow \mathbb{R}^+$ with the property that $\int q(u, w) dw = 1$ for every $u \in \mathbb{R}^\ell$. Recall the notation, introduced in Section 1.4.1, that we use the function $q(u, w)$ to denote a pdf and simultaneously, a probability measure $q(u, dw)$. We create a Markov chain $\{u^{(n)}\}_{n \in \mathbb{N}}$ that is invariant for ρ as follows. Define²

$$a(u, w) = 1 \wedge \frac{\rho(w)q(w, u)}{\rho(u)q(u, w)}. \quad (3.13)$$

The algorithm is as follows:

1. Set $n = 0$ and choose $u^{(0)} \in \mathbb{R}^\ell$.
2. $n \rightarrow n + 1$.
3. Draw $w^{(n)} \sim q(u^{(n-1)}, \cdot)$.
4. Set $u^{(n)} = w^{(n)}$ with probability $a(u^{(n-1)}, w^{(n)})$, $u^{(n)} = u^{(n-1)}$ otherwise.
5. Go to step 2.

At each step in the algorithm there are two sources of randomness: that required for drawing $w^{(n)}$ in step 3, and that required for accepting or rejecting $w^{(n)}$ as the next $u^{(n)}$ in step 4. These two sources of randomness are chosen to be independent of each other. Furthermore, all the randomness at discrete algorithmic time n is independent of randomness at preceding discrete algorithmic times, conditional on $u^{(n-1)}$. Thus the whole procedure gives a Markov chain. If $\mathbf{z} = \{\mathbf{z}^{(j)}\}_{j \in \mathbb{N}}$ is an i.i.d. sequence of $U[0, 1]$ random variables, then we may write the algorithm as follows:

$$\begin{aligned} w^{(n)} &\sim q(u^{(n-1)}, \cdot), \\ u^{(n)} &= w^{(n)} \mathbb{I}(\mathbf{z}^{(n)} \leq a(u^{(n-1)}, w^{(n)})) + u^{(n-1)} \mathbb{I}(\mathbf{z}^{(n)} > a(u^{(n-1)}, w^{(n)})). \end{aligned}$$

Here \mathbb{I} denotes the indicator function of a set. We let $p : \mathbb{R}^\ell \times \mathbb{R}^\ell \rightarrow \mathbb{R}^+$ denote the transition kernel of the resulting Markov chain, and we let p^n denote the transition kernel over n steps; recall that we therefore have $p^n(u, A) = \mathbb{P}(u^{(n)} \in A | u^{(0)} = u)$. Similarly as above, for fixed u , $p^n(u, dw)$ denotes a probability measure on \mathbb{R}^ℓ with density $p^n(u, w)$. The resulting algorithm is known as a Metropolis–Hastings MCMC algorithm, and it satisfies detailed balance with respect to μ .

Remark 3.5. *The following two observations are central to Metropolis–Hastings MCMC methods.*

² Recall that we use the \wedge operator to denote the minimum between two real numbers.

- The construction of Metropolis–Hastings MCMC methods is designed to ensure the detailed balance condition (3.12). We will use the condition expressed in this form in what follows later. It is also sometimes written in integrated form as the statement

$$\int_{\mathbb{R}^\ell \times \mathbb{R}^\ell} f(u, w) \rho(u) p(u, w) dudw = \int_{\mathbb{R}^\ell \times \mathbb{R}^\ell} f(u, w) \rho(w) p(w, u) dudw \quad (3.14)$$

for all $f : \mathbb{R}^\ell \times \mathbb{R}^\ell \rightarrow \mathbb{R}$. Once this condition is obtained, it follows trivially that the measure μ with density ρ is invariant, since for $f = f(w)$, we obtain

$$\begin{aligned} \int_{\mathbb{R}^\ell} f(w) \left(\int_{\mathbb{R}^\ell} \rho(u) p(u, w) du \right) dw &= \int_{\mathbb{R}^\ell} f(w) \rho(w) dw \int_{\mathbb{R}^\ell} p(w, u) du \\ &= \int_{\mathbb{R}^\ell} f(w) \rho(w) dw. \end{aligned}$$

Note that $\int_{\mathbb{R}^\ell} \rho(u) p(u, w) du$ is the density of the distribution of the Markov chain after one step, given that it is initially distributed according to density ρ . Thus the preceding identity shows that the expectation of f is unchanged by the Markov chain if it is initially distributed with density ρ . This means that if the Markov chain is distributed according to measure with density ρ initially, then it will be distributed according to the same measure for all algorithmic time.

- Note that in order to implement Metropolis–Hastings MCMC methods, it is not necessary to know the normalization constant for $\rho(\cdot)$, since only its ratio appears in the definition of the acceptance probability a .



The Metropolis–Hastings algorithm defined above satisfies the following corollary, which requires definition of total-variation (TV) distance given in Section 1.3:

Corollary 3.6. *For the Metropolis–Hastings MCMC methods, we have that the detailed balance condition (3.12) is satisfied and that hence μ is invariant: if $u^{(0)} \sim \mu$ with Lebesgue density ρ , then $u^{(n)} \sim \mu$ for all $n \in \mathbb{Z}^+$. Thus, if the Markov chain is ergodic, then the conclusions of Theorem 3.3 hold.*

We now describe some exemplars of Metropolis–Hastings methods adapted to the data assimilation problem. These are not to be taken as optimal MCMC methods for data assimilation, but rather as examples of how to construct proposal distributions $q(u, \cdot)$ for Metropolis–Hastings methods in the context of data assimilation. In any given application, the proposal distribution plays a central role in the efficiency of the MCMC method, and tailoring it to the specifics of the problem can have significant impact on efficiency of the MCMC method. Because of the level of generality at which we are presenting the material herein (arbitrary f and h), we cannot discuss such tailoring in any detail.

3.2.3. Deterministic Dynamics

In the case of deterministic dynamics (2.3), the measure of interest is a measure on the initial condition v_0 in \mathbb{R}^n . Perhaps the simplest Metropolis–Hastings algorithm is the **random walk Metropolis** (RWM) sampler, which employs a Gaussian proposal, centered at the current

state; we now illustrate this for the case of deterministic dynamics. Recall that the measure of interest is ν with pdf ϱ . Furthermore, $\varrho \propto \exp(-\mathbf{I}_{\text{det}}(v_0; y))$, as given in Theorem 2.11.

The RWM method proceeds as follows: Given that we are at $u^{(n-1)} \in \mathbb{R}^n$, a current approximate sample from the posterior distribution on the initial condition, we propose

$$w^{(n)} = u^{(n-1)} + \beta \iota^{(n-1)}, \quad (3.15)$$

where $\iota^{(n-1)} \sim N(0, C_{\text{prop}})$ for some symmetric positive definite proposal covariance C_{prop} and proposal variance scale parameter $\beta > 0$; natural choices for this proposal covariance include the identity I or the prior covariance C_0 . Because of the symmetry of such a random walk proposal, it follows that $q(w, u) = q(u, w)$ and hence that

$$\begin{aligned} a(u, w) &= 1 \wedge \frac{\varrho(w)}{\varrho(u)} \\ &= 1 \wedge \exp(\mathbf{I}_{\text{det}}(u; y) - \mathbf{I}_{\text{det}}(w; y)). \end{aligned}$$

Remark 3.7. *The expression for the acceptance probability shows that the proposed move to w is accepted with probability 1 if the value of $\mathbf{I}_{\text{det}}(\cdot; y)$, the log-posterior, is decreased by moving to w from the current state u . On the other hand, if $\mathbf{I}_{\text{det}}(\cdot; y)$ increases, then the proposed state is accepted only with some probability less than 1. Recall that $\mathbf{I}_{\text{det}}(\cdot; y)$ is the sum of the prior penalization (background) and the model–data misfit functional. The algorithm thus has a very natural interpretation in terms of the data-assimilation problem: it biases samples toward decreasing $\mathbf{I}_{\text{det}}(\cdot; y)$ and hence to improving the fit to both the model and the data in combination.*

The algorithm has two key tuning parameters: the proposal covariance C_{prop} and the scale parameter β . See Remark 3.4, first bullet, for discussion of the role of such parameters. The covariance can encode any knowledge, or guesses, about the relative strength of correlations in the model; given this, the parameter β should be tuned to give an acceptance probability that is neither close to 0 nor to 1. This is because if the acceptance probability is small, then successive states of the Markov chain are highly correlated, leading to a large constant K in (3.11). On the other hand, if the acceptance probability is close to 1, then this is typically because β is small, also leading to highly correlated steps and hence to a large constant K in (3.11).



Numerical results illustrating the method are given in Section 3.4.

3.2.4. Stochastic Dynamics

We now apply the Metropolis–Hastings methodology to the data assimilation smoothing problem in the case of the stochastic dynamics model (2.1). Thus the probability measure is on an entire signal sequence $\{v_j\}_{j=0}^J$ and not just on v_0 ; hence it lives on $\mathbb{R}^{|\mathbb{J}_0| \times n}$. It is possible to apply the random walk method to this situation, too, but we take the opportunity to introduce several different Metropolis–Hastings methods, in order to highlight the flexibility of the methodology. Furthermore, it is also possible to take the ideas behind the proposals introduced in this section and apply them in the case of deterministic dynamics.

In what follows, recall the measures μ_0 and μ defined in Section 2.3, with densities ρ_0 and ρ , representing (respectively) the measure on sequences v generated by (2.1) and the resulting measure when the signal is conditioned on the data y from (2.2). We now construct,

via the Metropolis–Hastings methodology, two Markov chains $\{u^{(n)}\}_{n \in \mathbb{N}}$ that are invariant with respect to μ . Hence we need only specify the transition kernel $q(u, w)$ and identify the resulting acceptance probability $a(u, w)$. The sequence $\{w^{(n)}\}_{n \in \mathbb{Z}^+}$ will denote the proposals.

Independence Dynamics Sampler Here we choose the proposal $w^{(n)}$, independently of the current state $u^{(n-1)}$, from the prior μ_0 with density ρ_0 . Thus we are simply proposing independent draws from the dynamical model (2.1), with no information from the data used in the proposal. Important in what follows is the observation that

$$\frac{\rho(v)}{\rho_0(v)} \propto \exp(-\Phi(v; y)). \quad (3.16)$$

With the given definition of proposal, we have that $q(u, w) = \rho_0(w)$ and hence that

$$\begin{aligned} a(u, w) &= 1 \wedge \frac{\rho(w)q(w, u)}{\rho(u)q(u, w)} \\ &= 1 \wedge \frac{\rho(w)/\rho_0(w)}{\rho(u)/\rho_0(u)} \\ &= 1 \wedge \exp(\Phi(u; y) - \Phi(w; y)). \end{aligned}$$

Remark 3.8. *The expression for the acceptance probability shows that the proposed move to w is accepted with probability 1 if the value of $\Phi(\cdot; y)$ is decreased by moving to w from the current state u . On the other hand, if $\Phi(\cdot; y)$ increases, then the proposed state is accepted only with some probability less than 1, with the probability decreasing exponentially fast with respect to the size of the increase. Recall that $\Phi(\cdot; y)$ measures the fit of the signal to the data. Because the proposal builds in the underlying signal model, the acceptance probability does not depend on $I(\cdot; y)$, the negative log-posterior, but only the part reflecting the data, namely the negative log-likelihood. In contrast, the RWM method, explained in the context of deterministic dynamics, does not build the model into its proposal and hence the accept–reject mechanism depends on the entire log-posterior; see Remark 3.7.* ♠

The independence dynamics sampler does not have any tuning parameters and hence can be very inefficient, since there are no parameters to modify in order to obtain a reasonable acceptance probability; as we will see in the illustrations in Section 3.4 below, the method can hence be quite inefficient because of the resulting frequent rejections. We now discuss this point and an approach to resolving it. The rejections are caused by attempts to move far from the current state, and in particular to proposed states that are based on the underlying stochastic dynamics, but not on the observed data. This typically leads to increases in the model–data misfit functional $\Phi(\cdot; y)$ once the Markov chain has found a state that fits the data reasonably well. Even if data is not explicitly used in constructing the proposal, this effect can be ameliorated by making local proposals, which do not move far from the current state. These are exemplified in the following MCMC algorithm.

The pCN Method. It is helpful in what follows to recall the measure ϑ_0 with density π_0 found from μ_0 and ρ_0 in the case $\Psi \equiv 0$ and given by equation (2.24). We denote the mean by m and covariance by C , noting that $m = (m_0^T, 0^T, \dots, 0^T)^T$ and that C is block diagonal with first block C_0 and the remainder all being Σ . Thus $\vartheta_0 = N(m, C)$. The basic idea of this method is to make proposals with the property that if $\Psi \equiv 0$, so that the dynamics is Gaussian and with no time correlation, and if $h \equiv 0$, so that the data is totally uninformative, then the proposal would be accepted with probability 1. Making small incremental proposals of this type then leads to a Markov chain that incorporates the effects of $\Psi \neq 0$ and $h \neq 0$ through the accept–reject mechanism. We describe the details of how this works.

Recall the prior on the stochastic dynamics model with density $\rho_0(v) \propto \exp(-\mathbf{J}(v))$ given by (2.19). It will be useful to rewrite π_0 as follows:

$$\pi_0(v) \propto \exp(-\mathbf{J}(v) + F(v)),$$

where

$$F(v) = \sum_{j=0}^{J-1} \left(\frac{1}{2} \left| \Sigma^{-\frac{1}{2}} \Psi(v_j) \right|^2 - \left\langle \Sigma^{-\frac{1}{2}} v_{j+1}, \Sigma^{-\frac{1}{2}} \Psi(v_j) \right\rangle \right). \quad (3.17)$$

We note that

$$\frac{\rho_0(v)}{\pi_0(v)} \propto \exp(-F(v))$$

and hence that using (2.22),

$$\frac{\rho(v)}{\pi_0(v)} \propto \exp(-\Phi(v; y) - F(v)). \quad (3.18)$$

Recall the Gaussian measure $\vartheta_0 = N(m, C)$ defined via its pdf in (2.24). The pCN method is a variant of random-walk-type methods, based on the following Gaussian proposal:

$$\begin{aligned} w^{(n)} &= m + (1 - \beta^2)^{\frac{1}{2}} (u^{(n-1)} - m) + \beta \iota^{(n-1)}, \\ \beta &\in (0, 1], \quad \iota^{(n-1)} \sim N(0, C). \end{aligned} \quad (3.19)$$

Here $\iota^{(n-1)}$ is assumed to be independent of $u^{(n-1)}$.

Lemma 3.9. *Consider the Markov chain*

$$\begin{aligned} u^{(n)} &= m + (1 - \beta^2)^{\frac{1}{2}} (u^{(n-1)} - m) + \beta \iota^{(n-1)}, \\ \beta &\in (0, 1], \quad \iota^{(n-1)} \sim N(0, C), \end{aligned} \quad (3.20)$$

with $\iota^{(n-1)}$ independent of $u^{(n-1)}$. The Markov kernel for this chain $q(u, w)$ satisfies detailed balance (3.12) with respect to the measure ϑ_0 with density π_0 :

$$\frac{\pi_0(w)q(w, u)}{\pi_0(u)q(u, w)} = 1. \quad (3.21)$$

Proof We show that $\pi_0(u)q(u, w)$ is symmetric in (u, w) . To demonstrate this, it suffices to consider the quadratic form found by taking the negative of the logarithm of this expression. This is given by

$$\frac{1}{2} |u - m|_C^2 + \frac{1}{2\beta^2} |w - m - (1 - \beta^2)^{\frac{1}{2}}(u - m)|_C^2.$$

This is the same as

$$\frac{1}{2\beta^2} |u - m|_C^2 + \frac{1}{2\beta^2} |w - m|_C^2 - \frac{(1 - \beta^2)^{\frac{1}{2}}}{\beta^2} \langle w - m, u - m \rangle_C,$$

which is clearly symmetric in (u, w) . The result follows. \square

By use of (3.21) and (3.18), we deduce that the acceptance probability for the MCMC method with proposal (3.19) is

$$\begin{aligned} a(u, w) &= 1 \wedge \frac{\rho(w)q(w, u)}{\rho(u)q(u, w)} \\ &= 1 \wedge \frac{\rho(w)/\pi_0(w)}{\rho(u)/\pi_0(u)} \\ &= 1 \wedge \exp(\Phi(u; y) - \Phi(w; y) + F(u) - F(w)). \end{aligned}$$

Recall that the proposal preserves the underlying Gaussian structure of the stochastic dynamics model; the accept–reject mechanism then introduces non-Gaussianity into the stochastic dynamics model, via F , and introduces the effect of the data, via Φ . By choosing β small, so that $w^{(n)}$ is close to $u^{(n-1)}$, we can make $a(v^{(n-1)}, w^{(n)})$ reasonably large and obtain a usable algorithm. This is illustrated in Section 3.4.

Recall from Section 2.3.3 that if $\Psi \equiv 0$ (as assumed to define the measure ϑ_0), then the noise sequence $\{\xi_{j-1}\}_{j=1}^\infty$ is identical with the signal sequence $\{v_j\}_{j=1}^\infty$. More generally, even if $\Psi \neq 0$, the noise sequence $\{\xi_j\}_{j=1}^\infty$, together with v_0 , a vector that we denote in Section 2.3.3 by ξ , uniquely determines the signal sequence $\{v_j\}_{j=0}^\infty$; see Lemma 2.9. This motivates a different formulation of the smoothing problem for stochastic dynamics in which one views the noise sequence and initial condition as the unknown, rather than the signal sequence itself. Here we study the implication of this perspective for MCMC methodology, in the context of the pCN method, leading to our third sampler within this subsection: the pCN dynamics sampler. We now describe this algorithm.

The pCN Dynamics Sampler is so named because the proposal (implicitly, via the mapping G defined in Lemma 2.9) samples from the dynamics as in the independence sampler, while the proposal also includes a parameter β allowing small steps to be taken and chosen to ensure good acceptance probability, as in the pCN method. The posterior measure we wish to sample is given in Theorem 2.10. Note that this theorem implicitly contains the fact that

$$\vartheta(d\xi) \propto \exp(-\Phi_R(\xi; y)) \vartheta_0(d\xi).$$

Furthermore, $\vartheta_0 = N(m, C)$, where the mean m and covariance C are as described above for the standard pCN method. We use the pCN proposal (3.19)

$$\zeta^{(n)} = m + (1 - \beta^2)^{\frac{1}{2}} (\xi^{(n-1)} - m) + \beta \iota^{(n-1)},$$

and the acceptance probability is given by

$$a(\xi, \zeta) = 1 \wedge \exp(\Phi_R(\xi; y) - \Phi_R(\zeta; y)).$$

In interpreting this formula, it is instructive to note that

$$\Phi_R(\xi; y) = \frac{1}{2} |y - G(\xi)|_{\Gamma_J}^2 = \frac{1}{2} \left| \Gamma_J^{-\frac{1}{2}} (y - G(\xi)) \right|^2 = \Phi(G(\xi); y),$$

and that ξ comprises both v_0 and the noise sequence $\{\xi_j\}_{j=0}^{J-1}$. Thus the method has the same acceptance probability as the independence dynamics sampler, albeit expressed in terms of initial condition and model noise rather than signal, and also possesses a tunable parameter β ; it thus has the nice conceptual interpretation of the acceptance probability that is present in the independence dynamics sampler, as well as the advantage of the pCN method that the proposal variance β may be chosen to ensure a reasonable acceptance probability.

3.3 Variational Methods

Sampling the posterior using MCMC methods can be prohibitively expensive. This is because in general, sampling involves generating many different points in the state space of the Markov chain. It can be of interest to generate a single point, or small number of points, that represent the salient features of the probability distribution, when this is possible. If the probability is peaked at one place or a small number of places, then simply locating these peaks may be sufficient in some applied contexts. This is the basis for variational methods that seek to maximize the posterior probability, thereby locating such peaks. In practice, this boils down to minimizing the negative log-posterior.

We begin by illustrating the idea in the context of the Gaussian distributions highlighted in Section 3.1 concerning the Kalman smoother. In the case of stochastic dynamics, Theorem 3.1 shows that $\mathbb{P}(v|y)$, the pdf of the posterior distribution, has the form

$$P(v|y) \propto \exp\left(-\frac{1}{2}|v - m|_L^2\right).$$

Now consider the problem

$$v^* = \operatorname{argmax}_{v \in \mathbb{R}^{|\mathcal{I}_0| \times n}} \mathbb{P}(v|y).$$

From the structure of $\mathbb{P}(v|y)$, we see that

$$v^* = \operatorname{argmin}_{v \in \mathbb{R}^{|\mathcal{I}_0| \times n}} I(v; y),$$

where

$$I(v; y) = \frac{1}{2}|v - m|_L^2 = \frac{1}{2}|L^{-\frac{1}{2}}(v - m)|^2.$$

Thus $v^* = m$, the mean of the posterior. Similarly, using Theorem 3.2, we can show that in the case of deterministic dynamics,

$$v_0^* = \operatorname{argmax}_{v_0 \in \mathbb{R}^{|\mathcal{I}_0| \times n}} \mathbb{P}(v_0|y),$$

is given by $v_0^* = m_{\text{det}}$.

In this section, we show how to characterize peaks in the posterior probability, in the general non-Gaussian case, leading to problems in the calculus of variations. The methods are termed **variational methods**. In the atmospheric sciences, these variational methods are referred to as **4DVAR**; this nomenclature reflects the fact that they are variational methods that incorporate data over three spatial dimensions and one temporal dimension (thus four dimensions in total) in order to estimate the state. In Bayesian statistics, the methods are called **MAP estimators**: maximum a posteriori estimators. It is helpful to realize that the MAP estimator is not, in general, equal to the mean of the posterior distribution. However, in the case of Gaussian posteriors, it is equal to the mean. Computation of the mean of a posterior distribution, in general, requires integrating against the posterior distribution. This can be achieved, via sampling for example, but is typically quite expensive if sampling is expensive. MAP estimators, in contrast, require only the solution of an optimization problem. Unlike the previous section on MCMC methods, we do not attempt to review the vast literature on relevant algorithms (here optimization algorithms); instead, references are given in the bibliographic notes of Section 3.5.

First we consider the case of stochastic dynamics.

Theorem 3.10. Consider the data-assimilation problem for stochastic dynamics: (2.1), (2.2), with $\Psi \in C^1(\mathbb{R}^n, \mathbb{R}^n)$ and $h \in C^1(\mathbb{R}^n, \mathbb{R}^m)$. Then:

- (i) The infimum of $I(\cdot; y)$ given in (2.21) is attained at at least one point v^* in $\mathbb{R}^{|\mathbb{J}_0| \times n}$. It follows that the density $\rho(v) = \mathbb{P}(v|y)$ on $\mathbb{R}^{|\mathbb{J}_0| \times n}$ associated with the posterior probability μ given by Theorem 2.8 is maximized at v^* .
- (ii) Furthermore, let $B(u, \delta)$ denote a ball in $\mathbb{R}^{|\mathbb{J}_0| \times n}$ of radius δ and centered at u . Then

$$\lim_{\delta \rightarrow 0} \frac{\mathbb{P}^\mu(B(u_1, \delta))}{\mathbb{P}^\mu(B(u_2, \delta))} = \exp(I(u_2; y) - I(u_1; y)) \quad \text{for all } u_1, u_2 \in \mathbb{R}^{|\mathbb{J}_0| \times n}. \quad (3.22)$$

Proof Note that $I(\cdot; y)$ is nonnegative and continuous, so that the infimum \bar{I} is finite and nonnegative. To show that the infimum of $I(\cdot; y)$ is attained in $\mathbb{R}^{|\mathbb{J}_0| \times n}$, we let $v^{(n)}$ denote a minimizing sequence. Without loss of generality, we may assume that for all $n \in \mathbb{N}$,

$$I(v^{(n)}; y) \leq \bar{I} + 1.$$

From the structure of $I(\cdot; y)$, it follows that

$$\begin{aligned} v_0 &= m_0 + C_0^{\frac{1}{2}} r_0, \\ v_{j+1} &= \Psi(v_j) + \Sigma^{\frac{1}{2}} r_{j+1}, \quad j \in \mathbb{Z}^+, \end{aligned}$$

where $\frac{1}{2}|r_j|^2 \leq \bar{I} + 1$ for all $j \in \mathbb{Z}^+$. By iterating and using the inequalities on the $|r_j|$, we deduce the existence of $K > 0$ such that $|v^{(n)}| \leq K$ for all $n \in \mathbb{N}$. From this bounded sequence, we may extract a convergent subsequence, relabeled $v^{(n)}$ for simplicity, with limit v^* . By construction, we have that $v^{(n)} \rightarrow v^*$ and that for every $\epsilon > 0$, there is $N = N(\epsilon)$ such that

$$\bar{I} \leq I(v^{(n)}; y) \leq \bar{I} + \epsilon, \quad \forall n \geq N.$$

Hence, by continuity of $I(\cdot; y)$, it follows that

$$\bar{I} \leq I(v^*; y) \leq \bar{I} + \epsilon.$$

Since $\epsilon > 0$ is arbitrary, it follows that $I(v^*; y) = \bar{I}$. Because

$$\begin{aligned} \mu(dv) &= \frac{1}{Z} \exp(-I(v; y)) dv \\ &= \rho(v) dv, \end{aligned}$$

it follows that v^* also maximizes the posterior pdf ρ .

For the final result, we first note that because Ψ and h are continuously differentiable, the function $I(\cdot; y)$ is continuously differentiable. Thus we have

$$\begin{aligned} \mathbb{P}^\mu(B(u, \delta)) &= \frac{1}{Z} \int_{|v-u|<\delta} \exp(-I(v; y)) dv \\ &= \frac{1}{Z} \int_{|v-u|<\delta} \left(\exp(-I(u; y)) + e(u; v - u) \right) dv, \end{aligned}$$

where

$$e(u; v - u) = \left\langle - \int_0^1 D_v I(u + s(v - u); y) ds, v - u \right\rangle.$$

As a consequence, we have, for $K^\pm > 0$,

$$-K^-|\delta| \leq e(u; v - u) \leq K^+|\delta|$$

for $u = u_1, u_2$ and $|v - u| < \delta$. Using the preceding, we find that for $E := \exp(I(u_2; y) - I(u_1; y))$,

$$\frac{\mathbb{P}^\mu(B(u_1, \delta))}{\mathbb{P}^\mu(B(u_2, \delta))} \leq E \frac{\int_{|v-u_1|<\delta} \exp(K^+|\delta|) dv}{\int_{|v-u_2|<\delta} \exp(-K^-|\delta|) dv} = E \frac{\exp(K^+|\delta|)}{\exp(-K^-|\delta|)}.$$

Similarly, we have that

$$\frac{\mathbb{P}^\mu(B(u_1, \delta))}{\mathbb{P}^\mu(B(u_2, \delta))} \geq E \frac{\int_{|v-u_1|<\delta} \exp(-K^-|\delta|) dv}{\int_{|v-u_2|<\delta} \exp(K^+|\delta|) dv} = E \frac{\exp(-K^-|\delta|)}{\exp(K^+|\delta|)}.$$

Taking the limit $\delta \rightarrow 0$ gives the desired result. \square

Remark 3.11. The second statement in Theorem 3.10 may appear a little abstract. It is, however, essentially a complicated way of restating the first statement. To see this, fix u_2 and note that the right-hand side of (3.22) is maximized at point u_1 , which minimizes $I(\cdot; y)$. Thus, independently of the choice of fixed u_2 , the identity (3.22) shows that the probability of a small ball of radius δ centered at u_1 is approximately maximized by choosing centers at minimizers of $I(\cdot; y)$. Why, then, do we bother with the second statement? We do so because it makes no reference to Lebesgue density. As such, it can be generalized to infinite dimensions, as is required in continuous time, for example. We include the second statement for precisely this reason. We also remark that our assumption on continuous differentiability of Ψ and h is stronger than what is needed, but makes for the rather explicit bounds used in the preceding proof and is hence pedagogically desirable. ♠

The preceding theorem leads to a natural algorithm: compute

$$v = \operatorname{argmin}_{u \in \mathbb{R}^{|\mathbb{J}_0| \times n}} I(u; y).$$

In applications to meteorology, this algorithm is known as **weak constraint 4DVAR**, and we denote this by **w4DVAR** in what follows. The word “weak” in this context is used to indicate that the deterministic dynamics model (2.3a) is not imposed as a strong constraint. Instead, the objective functional $I(\cdot; y)$ is minimized; this penalizes deviations from exact satisfaction of the deterministic dynamics model, as well as deviations from the data.

The w4DVAR method generalizes the standard **4DVAR** method, which may be derived from w4DVAR in the limit $\Sigma \rightarrow 0$, so that the prior on the model dynamics (2.1) is deterministic, but with a random initial condition, as in (2.3). In this case, the appropriate minimization is of $I_{\text{det}}(v_0; y)$, given by (2.29). This has the advantage of being a lower-dimensional minimization problem than w4DVAR; however, it is often a harder minimization problem, especially when the dynamics is chaotic. The basic 4DVAR algorithm is sometimes called **strong constraint 4DVAR** to denote the fact that the dynamics model (2.3a) is imposed as a strong constraint on the minimization of the model–data misfit with respect to the initial condition; we simply refer to the method as 4DVAR. The following theorem may be proved similarly to Theorem 3.10.

Theorem 3.12. Consider the data-assimilation problem for deterministic dynamics: (2.3), (2.2) with $\Psi \in C^1(\mathbb{R}^n, \mathbb{R}^n)$ and $h \in C^1(\mathbb{R}^n, \mathbb{R}^m)$. Then:

- (i) The infimum of $I_{\text{det}}(\cdot; y)$ given in (2.29) is attained at at least one point v_0^* in \mathbb{R}^n . It follows that the density $\varrho(v_0) = \mathbb{P}(v_0|y)$ on \mathbb{R}^n associated with the posterior probability ν given by Theorem 2.11 is maximized at v_0^* .
- (ii) Furthermore, if $B(z, \delta)$ denotes a ball in \mathbb{R}^n of radius δ centered at z , then

$$\lim_{\delta \rightarrow 0} \frac{\mathbb{P}^\nu(B(z_1, \delta))}{\mathbb{P}^\nu(B(z_2, \delta))} = \exp(I_{\text{det}}(z_2; y) - I_{\text{det}}(z_1; y)).$$

As in the case of stochastic dynamics, we do not discuss optimization methods to perform minimization associated with variational problems; this is because optimization is a well-established and mature research area that is hard to do justice to within the confines of this book. However, we conclude this section with an example that illustrates certain advantages of the Bayesian perspective over the optimization or variational perspective. Recall from Theorem 2.15 that the Bayesian posterior distribution is continuous with respect to small changes in the data. In contrast, computation of the global maximizer of the probability may be discontinuous as a function of data. To illustrate this, consider the probability measure μ^ϵ on \mathbb{R} with Lebesgue density proportional to $\exp(-V^\epsilon(u))$, where

$$V^\epsilon(u) = \frac{1}{4}(1-u^2)^2 + \epsilon u. \quad (3.23)$$

It is a straightforward application of the methodology behind the proof of Theorem 2.15 to show that μ^ϵ is Lipschitz continuous in ϵ , with respect to the Hellinger metric. Furthermore, the methodology behind Theorems 3.10 and 3.12 shows that the probability with respect to this measure is maximized whenever V^ϵ is minimized. The global minimum, however, changes discontinuously, even though the posterior distribution changes smoothly. This is illustrated in Figure 3.1, where the left-hand panel shows the continuous evolution of the probability density function, while the right-hand panel shows the discontinuity in the global maximizer of the probability (minimizer of V^ϵ) as ϵ passes through zero. The explanation for this difference between the fully Bayesian approach and MAP estimation is as follows. The measure μ^ϵ has two peaks, for small ϵ , close to ± 1 . The Bayesian approach accounts for both of these peaks simultaneously and weights their contribution to expectations. In contrast, the MAP estimation approach leads to a global minimum located near $u = -1$ for $\epsilon > 0$ and near $u = +1$ for $\epsilon < 0$, resulting in a discontinuity.

3.4 Illustrations

We describe a range of numerical experiments that illustrate the application of MCMC methods and variational methods to the smoothing problems that arise in both deterministic and stochastic dynamics.

The first illustration concerns the use of the RWM algorithm to study the smoothing distribution for Example 2.4 in the case of deterministic dynamics, where our aim is to find $\mathbb{P}(v_0|y)$. Recall Figure 2.13a, which shows the true posterior pdf, found by plotting the formula given in Theorem 2.8. We now approximate the true posterior pdf by the MCMC method, using the same parameters, namely $m_0 = 0.5$, $C_0 = 0.01$, $\gamma = 0.2$, and $v_0^\dagger = 0.3$. In Figure 3.2, we compare the posterior pdf calculated by the RWM method (denoted by ρ^N , the histogram

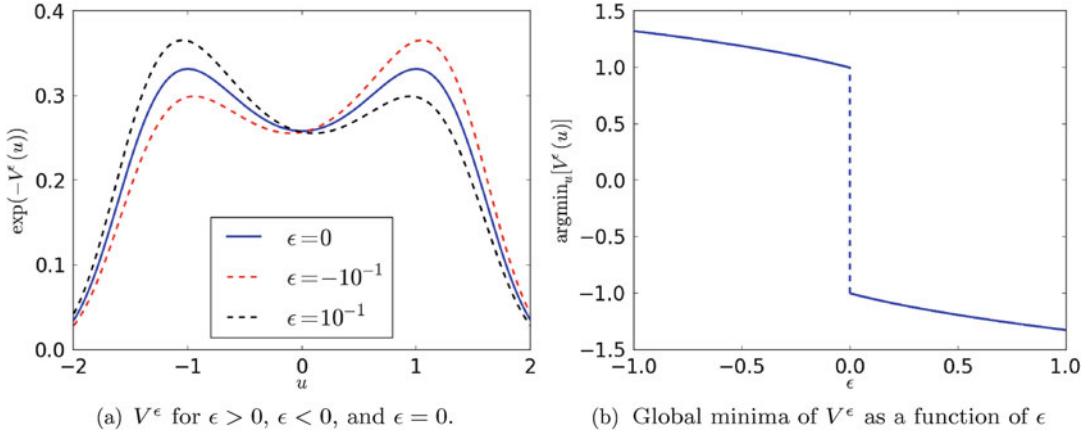


Fig. 3.1: Plot of (3.23) shows discontinuity of the global maximum as a function of ϵ .

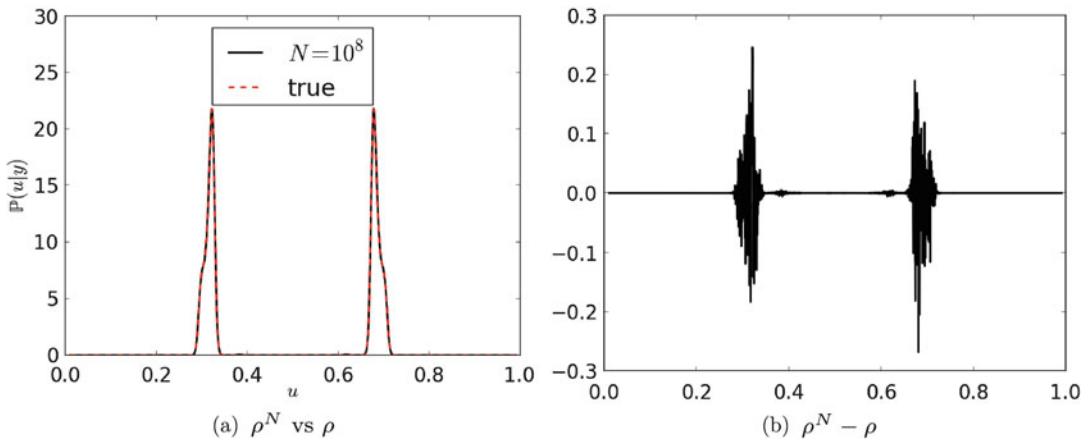


Fig. 3.2: Comparison of the posterior for Example 2.4 for $r = 4$ using random walk metropolis and equation (2.29) directly as in the MATLAB program p2.m. We have used $J = 5$, $C_0 = 0.01$, $m_0 = 0.5$, $\gamma = 0.2$, and true initial condition $v_0 = 0.3$; see also p3.m in Section 5.2.1. We have used $N = 10^8$ samples from the MCMC algorithm.

of the output of the Markov chain) with the true posterior pdf ρ . The two distributions are almost indistinguishable when plotted together in Figure 3.2a; in Figure 3.2b, we plot their difference, which, as we can see, is small relative to the true value. We deduce that the number of samples used, $N = 10^8$, results here in an accurate sampling of the posterior.

We now turn to the use of MCMC methods to sample the smoothing pdf $\mathbb{P}(v|y)$ in the case of stochastic dynamics (2.1), using the independence dynamics sampler and both pCN methods. Before describing the application of numerical methods, we study the ergodicity of the independence dynamics sampler in a simple but illustrative setting. For simplicity, assume that the observation operator h is bounded, so that for all $u \in \mathbb{R}^N$, $|h(u)| \leq h_{\max}$. Then, recalling the notation $Y_j = \{y_\ell\}_{\ell=1}^j$ from the filtering problem, we have

$$\begin{aligned}
\Phi(u; y) &\leq \sum_{j=0}^{J-1} (|\Gamma^{-\frac{1}{2}} y_{j+1}|^2 + |\Gamma^{-\frac{1}{2}} h(u_{j+1})|^2) \\
&\leq |\Gamma^{-\frac{1}{2}}|^2 \left(\sum_{j=0}^{J-1} |y_{j+1}|^2 + J h_{\max}^2 \right) \\
&\leq |\Gamma^{-\frac{1}{2}}|^2 (|Y_J|^2 + J h_{\max}^2) \\
&=: \Phi_{\max}.
\end{aligned}$$

Since $\Phi \geq 0$, this shows that every proposed step is accepted with probability exceeding $e^{-\Phi_{\max}}$, and hence that since proposals are made with the prior measure μ_0 describing the unobserved stochastic dynamics,

$$p(u, A) \geq e^{-\Phi_{\max}} \mu_0(A).$$

Thus Theorem 3.3 applies, and in particular, (3.10) and (3.11) hold, with $\varepsilon = e^{-\Phi_{\max}}$, under these assumptions. This positive result about the ergodicity of the MCMC method also indicates the potential difficulties with the independence dynamics sampler. The independence sampler relies on draws from the prior matching the data well. Where the data set is large ($J \gg 1$) or the noise covariance small ($|\Gamma| \ll 1$), this will happen infrequently, because Φ_{\max} will be large, and the MCMC method will reject frequently and be inefficient. To illustrate this, we consider application of the method to Example 2.3, using the same parameters as in Figure 2.3; specifically, we take $\alpha = 2.5$ and $\Sigma = \sigma^2 = 1$. We now sample the posterior distribution and then plot the resulting accept–reject ratio a for the independence dynamics sampler, employing different values of noise Γ and different sizes of the data set J . This is illustrated in Figure 3.3.

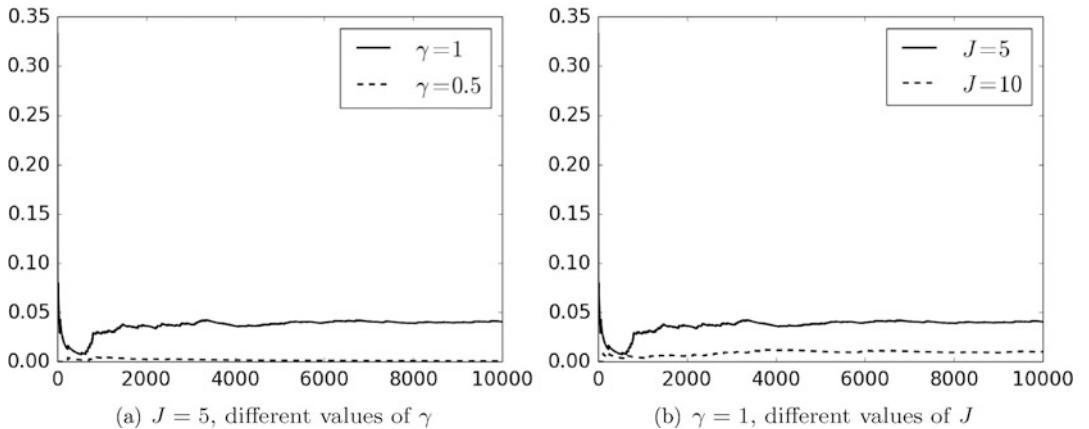


Fig. 3.3: Accept–reject probability of the independence sampler for Example 2.3 for $\alpha = 2.5$, $\Sigma = \sigma^2 = 1$, and $\Gamma = \gamma^2$ for different values of γ and J .

In addition, in Figure 3.4, we plot the output and the running average of the output projected into the first element of the vector $v^{(k)}$, the initial condition—recall that we are defining a Markov chain on \mathbb{R}^{J+1} —for $N = 10^5$ steps. Figure 3.4a clearly exhibits the fact that there are many rejections caused by the low average acceptance probability. Figure 3.4b

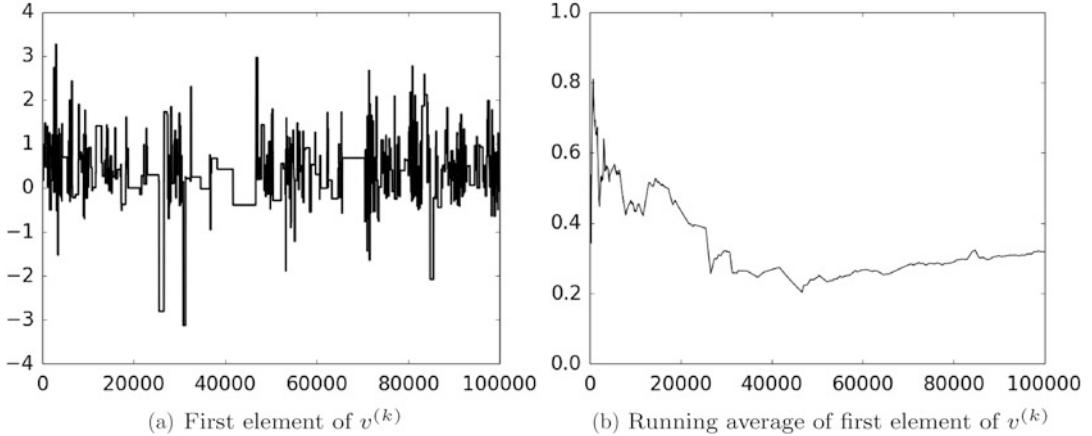


Fig. 3.4: Output and running average of the independence dynamics sampler after $K = 10^5$ steps, for Example 2.3 for $\alpha = 2.5$, $\Sigma = \sigma^2 = 1$, and $\Gamma = \gamma^2 = 1$, with $J = 10$; see also p4.m in Section 5.2.2.

shows that the running average has not converged after 10^5 steps, indicating that the chains needs to be run for longer. If we run the Markov chain over $N = 10^8$ steps, then we do get convergence. This is illustrated in Figure 3.5. In Figure 3.5a, we see that the running average has converged to its limiting value when this many steps are used. In Figure 3.5b, we plot the marginal probability distribution for the first element of $v^{(k)}$, calculated from this converged Markov chain.

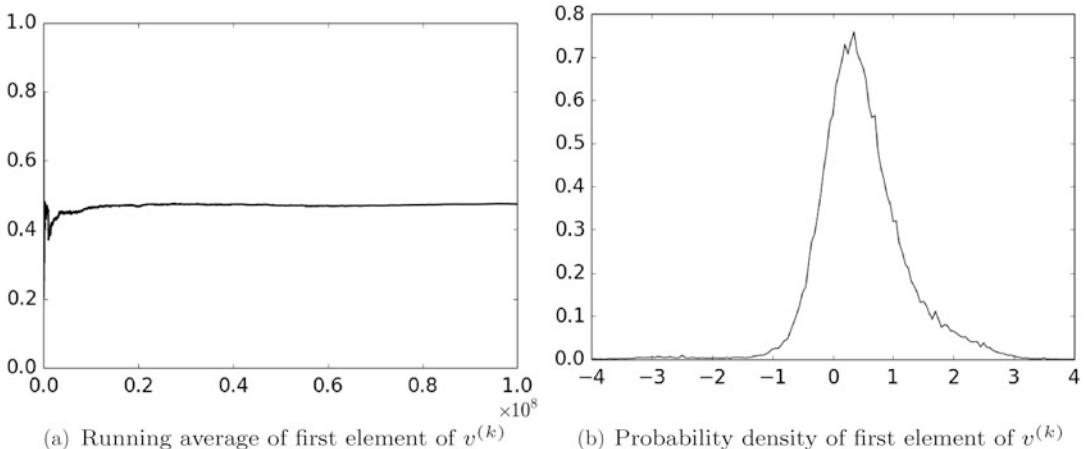


Fig. 3.5: Running average and probability density of the first element of $v^{(k)}$ for the independence dynamics sampler after $K = 10^8$ steps, for Example 2.3 for $\alpha = 2.5$, $\Sigma = \sigma^2 = 1$, and $\Gamma = \gamma^2$, with $\gamma = 1$ and $J = 10$; see also p4.m in Section 5.2.2.

In order to get faster convergence when sampling the posterior distribution, we turn to application of the pCN method. Unlike the independence dynamics sampler, this contains a tunable parameter that can vary the size of the proposals. In particular, the possibility of

making small moves, with resultant higher acceptance probability, makes this a more flexible method than the independence dynamics sampler. In Figure 3.6, we show application of the pCN sampler, again considering Example 2.3 for $\alpha = 2.5$, $\Sigma = \sigma^2 = 1$, and $\Gamma = \gamma^2 = 1$, with $J = 10$, the same parameters used in Figure 3.4.

In the case that the dynamics significantly influence the trajectory, i.e., the regime of large Ψ or small σ , it may be the case that the standard pCN method is ineffective, due to large effects of the G term, and the improbability of Gaussian samples being close to samples of the prior on the dynamics. The pCN dynamics sampler, recall, acts on the space comprising the initial condition and forcing, both of which are Gaussian under the prior, and so may sometimes have an advantage given that pCN-type methods are based on Gaussian proposals. The use of this method is explored in Figure 3.7 for Example 2.3 for $\alpha = 2.5$, $\Sigma = \sigma^2 = 1$, and $\Gamma = \gamma^2 = 1$, with $J = 10$.

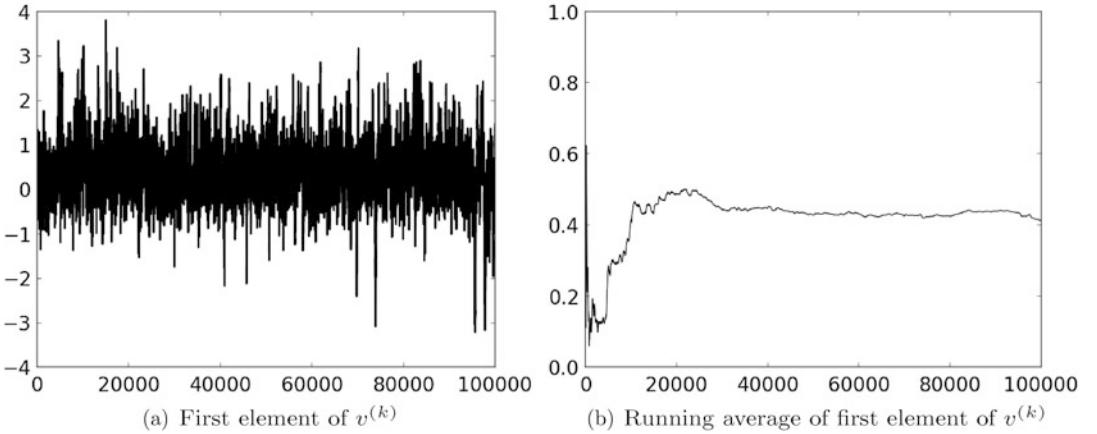


Fig. 3.6: Trace plot and running average of the first element of $v^{(k)}$ for the pCN sampler after $K = 10^5$ steps, for Example 2.3 with $\alpha = 2.5$, $\Sigma = \sigma^2 = 1$, and $\Gamma = \gamma^2 = 1$, with $J = 10$; see also p5.m in Section 5.2.3.

We now turn to variational methods; recall Theorems 3.10 and 3.12 in the stochastic and deterministic cases respectively. In Figure 3.8a, we plot the MAP (4DVAR) estimator for our Example 2.1, choosing exactly the same parameters and data as for Figure 2.10a, in the case $J = 10^2$. In this case, the function $I_{\text{det}}(\cdot; y)$ is quadratic and has a unique global minimum. A straightforward minimization routine will easily find this: we employed standard MATLAB optimization software initialized at three different points. From all three starting points chosen, the algorithm finds the correct global minimizer.

In Figure 3.8b, we plot the MAP (4DVAR) estimator for our Example 2.4 for the case $r = 4$ choosing exactly the same parameters and data as for Figure 2.13. We again employ a MATLAB optimization routine, and we again initialize it at three different points. The value obtained for our MAP estimator depends crucially on the choice of initial condition in our minimization procedure, in particular on the choices of starting point presented: for the three initializations shown, it is only when we start from 0.2 that we are able to find the global minimum of $I_{\text{det}}(v_0; y)$. By Theorem 3.12, this global minimum corresponds to the maximum of the posterior distribution, and we see that finding the MAP estimator is a difficult task for this problem. Starting with the other two initial conditions displayed, we converge to one of the many local minima of $I_{\text{det}}(v_0; y)$; these local minima are in fact regions of very low probability,

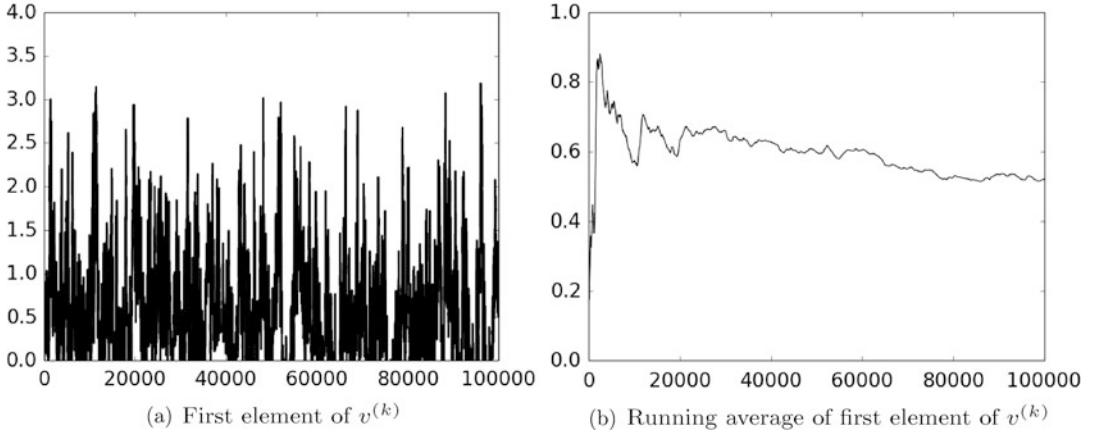


Fig. 3.7: Trace plot and running average of the first element of $v^{(k)}$ for the pCN dynamics sampler after $K = 10^5$ steps, for Example 2.3 with $\alpha = 2.5$, $\Sigma = \sigma^2 = 1$, and $\Gamma = \gamma^2 = 1$, with $J = 10$; see also `p6.m` in Section 5.2.3.

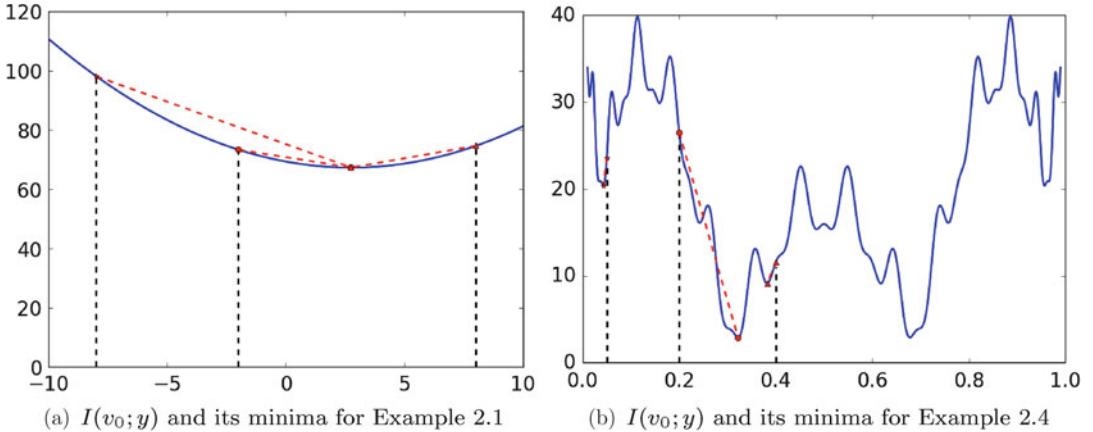


Fig. 3.8: Finding local minima of $I(v_0; y)$ for Examples 2.1 and 2.4. The values and the data used are the same as for Figures 2.10a and 2.13b. (\circ , \star , \square) denote three different initial conditions for starting the minimization process: $(-8, -2, 8)$ for Example 2.1 and $(0.05, 0.2, 0.4)$ for Example 2.4.

as we can see in Figure 2.13a. This illustrates the care required in computing 4DVAR solutions in cases in which the forward problem exhibits sensitivity to initial conditions.

Figure 3.9 shows application of the w4DVAR method, or MAP estimator given by Theorem 3.10, in the case of Example 2.3 with parameters set at $J = 5$, $\gamma = \sigma = 0.1$. In contrast to the previous example, this is no longer a one-dimensional minimization problem: we are minimizing $I(v; y)$ given by (2.21) over $v \in \mathbb{R}^6$, given the data $y \in \mathbb{R}^5$. The figure shows that there are at least two local minimizers for this problem, with $v^{(1)}$ closer to the truth than $v^{(2)}$, and with $I(v^{(1)}; y)$ considerably smaller than $I(v^{(2)}; y)$. However, $v^{(2)}$ has a larger basin of attraction for the optimization software used: many initial conditions lead to $v^{(2)}$, while fewer lead to $v^{(1)}$. Furthermore, while we believe that $v^{(1)}$ is the global minimizer, it is difficult

to state this with certainty, even for this relatively low-dimensional model. To get greater certainty, an exhaustive and expensive search of the six-dimensional parameter space would be needed.

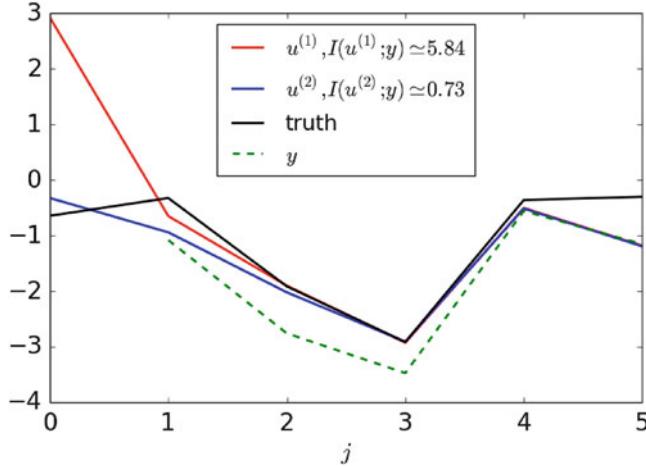


Fig. 3.9: Weak constraint 4DVAR for $J = 5$, $\gamma = \sigma = 0.1$, illustrating two local minimizers $v^{(1)}$ and $v^{(2)}$; see also p7.m in Section 5.2.5.

3.5 Bibliographic Notes

- The Kalman smoother from Section 3.1 leads to a system of linear equations, characterized in Theorem 3.1. These equations are of block tridiagonal form, and may be solved by LU factorization. The Kalman filter corresponds to the LU sweep in this factorization, a fact that was highlighted in [26].
- Section 3.2. Monte Carlo Markov chain methods have a long history, initiated in the 1953 paper [107] and then generalized to an abstract formulation in the 1970 paper [66]. The subject is presented from an algorithmic point of view in [92]. Theorem 3.3 is contained in [108], and that reference also contains many other convergence theorems for Markov chains; in particular, we note that it is often possible to increase substantially the class of functions φ to which the theorem applies by means of Lyapunov function techniques, which control the tails of the probability distribution. The specific form of the pCN-MCMC method that we introduce here has been chosen to be particularly effective in high dimensions; see [35] for an overview, [18] for the introduction of pCN and other methods for sampling probability measures in infinite dimensions in the context of conditioned diffusions, and [34] for an application to a data-assimilation problem.

The key point about pCN methods is that the proposal is reversible with respect to an underlying Gaussian measure. Even in the absence of data, if $\Psi \neq 0$, then this Gaussian measure is far from the measure governing the actual dynamics. In contrast, still in the absence of data, this Gaussian measure is *precisely* the measure governing the noise and initial condition, giving the pCN dynamics sampler a natural advantage over the standard pCN method. In particular, notice that the acceptance probability is now determined only

by the model–data misfit for the pCN dynamics sampler, and does *not* have to account for incorporation of the dynamics as it does in the original pCN method; this typically improves the acceptance rate of the pCN dynamics sampler over the standard pCN method. Therefore, this method may be preferable, particularly in the case of unstable dynamics. The pCN dynamics sampler was introduced in [34] and further tested in [69]; it shows considerable promise.

The subject of MCMC methods is an enormous one, to which we cannot do justice in this brief presentation. There are two relevant time scales for the Markov chain: the burn-in time, which determines the time to reach the part of state-space where most of the probability mass is concentrated, and the mixing time, which determines the time taken to fully explore the probability distribution. Our brief overview would not be complete without a cursory discussion of convergence diagnostics [54], which attempt to ensure that the Markov chain is run long enough to have both burnt in and mixed. While none of the diagnostics are foolproof, there are many simple tests that can and should be undertaken. The first is simply to study (as we have done in this section) trace plots of quantities of interest (components of the solution, acceptance probabilities) and the running averages of these quantities of interest. More sophisticated diagnostics are also available. For example, comparison of the within-chain and between-chain variances of multiple chains beginning from overdispersed initial conditions is advocated in the works [55, 25]. The authors of those works advise that one apply a range of tests based on comparing inferences from individual chains and a mixture of chains. These and other more sophisticated diagnostics are not considered further here, and the reader is referred to the cited works for further details and discussion.

- Section 3.3. Variational methods, known as 4DVAR in the meteorology community and widely used in practice, have the distinction, when compared with the ad hoc non-Gaussian filters described in the next chapter, which are also widely used in practice in their EnKF and 3DVAR formulations, of being well founded statistically: they correspond to the maximum a posteriori estimator (MAP estimator) for the fully Bayesian posterior distribution on model state given data [78]. See [151] and the references therein for a discussion of the applied context; see [43] for a more theoretical presentation, including connections to the Onsager–Machlup functional arising in the theory of diffusion processes. The European Centre for Medium-Range Weather Forecasts (ECMWF) runs a weather prediction code based on spectral approximation of continuum versions of Newton’s balance laws, together with various subgrid scale models. Initialization of this prediction code is based on the use of 4DVAR-like methods. The conjunction of this computational forward model and the use of 4DVAR to incorporate data results in the best weather predictor worldwide, according to a widely adopted metric by which the prediction skill of forecasts is measured. The subject of algorithms for optimization, which of course underpins variational methods, is vast, and we have not attempted to cover it here; we mention briefly that many methods use first-derivative information (for example steepest-descent methods) and second-derivative information (Newton methods); the reader is directed to [112] for details. Derivatives can also be useful in making MCMC proposals, leading to the Langevin and the hybrid Monte Carlo methods, for example; see [124] and the references therein.

3.6 Exercises

1. Consider the posterior distribution on the initial condition, given by Theorem 2.11, in the case of deterministic dynamics. In the case of Example 2.4, program p2.m plots the prior and posterior distributions for this problem for data generated with true initial condition $v_0 = 0.1$. Why is the posterior distribution concentrating much closer to 0.9 than to the true initial condition at 0.1? Change the mean of the prior from 0.7 to 0.3. What do you observe regarding the effect on the posterior? Explain what you observe. Illustrate your findings with graphics.
2. Consider the posterior distribution on the initial condition, given by Theorem 2.11, in the case of deterministic dynamics. In the case of Example 2.4, program p3.m approximates the posterior distribution for this problem for data generated with true initial condition $v_0 = 0.3$. Why is the posterior distribution in this case approximately symmetric about 0.5? What happens if the mean of the prior is changed from 0.5 to 0.1? Explain what you observe. Illustrate your findings with graphics.
3. Consider the posterior distribution on the initial condition, given by Theorem 2.11, in the case of deterministic dynamics. In the case of Example 2.4, program p3.m approximates the posterior distribution for this problem. Modify the program so that the prior and data are the same as for the first exercise in this section. Compare the approximation to the posterior obtained by use of program p3.m with the true posterior as computed by program p2.m. Carry out similar comparisons for different choices of prior, ensuring that programs p2.m and p3.m share the same prior and the same data. In all cases, experiment with the choice of the parameter β in the proposal distribution within p3.m, and determine its effect on the displayed approximation of the true posterior computed from p2.m. Illustrate your findings with graphics.
4. Consider the posterior distribution on the initial condition, given by Theorem 2.11, in the case of deterministic dynamics. In the case of Example 2.4, program p3.m approximates the posterior distribution for this problem. Modify the program so that it applies to Example 2.3. Experiment with the choice of the parameter J , which determines the length of the Markov chain simulation, within p3.m. Illustrate your findings with graphics.
5. Consider the posterior distribution on the signal, given by Theorem 2.8, in the case of stochastic dynamics. In the case of Example 2.3, program p4.m approximates the posterior distribution for this problem, using the independence dynamics sampler. Run this program for a range of values of γ . Report and explain the effect of γ on the acceptance probability curves.
6. Consider the posterior distribution on the signal, given by Theorem 2.8, in the case of stochastic dynamics. In the case of Example 2.3, program p5.m approximates the posterior distribution for this problem, using the pCN sampler. Run this program for a range of values of γ . Report and explain the effect of β on the acceptance probability curves.
7. Consider the posterior distribution on the signal, given by Theorem 2.8, in the case of stochastic dynamics. In the case of Example 2.3, program p6.m approximates the posterior distribution for this problem, using the pCN dynamics sampler. Run this program for a range of values of γ . Report and explain the effect of σ and of J on the acceptance probability curves.
8. Consider the MAP estimator for the posterior distribution on the signal, given by Theorem 3.10, in the case of stochastic dynamics. Program p7.m finds the MAP estimator for Example 2.3. Increase J to 50 and display your results graphically. Now repeat

your experiments for the values $\gamma = 0.01, 0.1$, and 10 and display and discuss your findings. Repeat the experiments using the “truth” as the initial condition for the minimization. What effect does this have? Explain this effect.

9. Prove Theorem 3.12.
10. Consider application of the RWM proposal (3.15), applied in the case of stochastic dynamics. Find the form of the Metropolis–Hastings acceptance probability in this case.
11. Consider the family of probability measures μ^ϵ on \mathbb{R} with Lebesgue density proportional to $\exp(-V^\epsilon(u))$ with $V^\epsilon(u)$ given by (3.23). Prove that the family of measure μ^ϵ is locally Lipschitz in the Hellinger metric and in the total variation metric.

Chapter 4

Discrete Time: Filtering Algorithms

In this chapter, we describe various algorithms for the filtering problem. Recall from Section 2.4 that filtering refers to the sequential update of the probability distribution on the state given the data, as data is acquired, and that $Y_j = \{y_\ell\}_{\ell=1}^j$ denotes the data accumulated up to time j . The filtering update from time j to time $j + 1$ may be broken into two steps: *prediction*, which is based on the equation for the state evolution, using the Markov kernel for the stochastic or deterministic dynamical system that maps $\mathbb{P}(v_j|Y_j)$ into $\mathbb{P}(v_{j+1}|Y_j)$; and *analysis*, which incorporates data via Bayes's formula and maps $\mathbb{P}(v_{j+1}|Y_j)$ into $\mathbb{P}(v_{j+1}|Y_{j+1})$. All but one of the algorithms we study (the optimal proposal version of the particle filter) will also reflect these two steps.

We begin in Section 4.1 with the Kalman filter, which provides an exact algorithm to determine the filtering distribution for linear problems with additive Gaussian noise. Since the filtering distribution is Gaussian in this case, the algorithm comprises an iteration that maps the mean and covariance from time j to time $j + 1$. In Section 4.2, we show how the idea of Kalman filtering may be used to combine a dynamical model with data for nonlinear problems; in this case, the posterior distribution is not Gaussian, but the algorithms proceed by invoking a Gaussian ansatz in the analysis step of the filter. This results in algorithms that do not provably approximate the true filtering distribution in general; in various forms, they are, however, robust to use in high dimensions. In Section 4.3, we introduce the particle filter methodology, which leads to provably accurate estimates of the true filtering distribution but which is, in its current forms, poorly behaved in high dimensions. The algorithms in Sections 4.1–4.3 are concerned primarily with stochastic dynamics, but setting $\Sigma = 0$ yields the corresponding algorithms for deterministic dynamics. In Section 4.4, we study the long-time behavior of some of the filtering algorithms introduced in the previous sections. Finally, in Section 4.5, we present some numerical illustrations and conclude with bibliographic notes and exercises in Sections 4.6 and 4.7.

For clarity of exposition, we again recall the form of the data assimilation problem. The signal is governed by the model of equations (2.1):

$$v_{j+1} = \Psi(v_j) + \xi_j, \quad j \in \mathbb{Z}^+,$$

$$v_0 \sim N(m_0, C_0),$$

Electronic supplementary material The online version of this chapter (doi: [10.1007/978-3-319-20325-6_4](https://doi.org/10.1007/978-3-319-20325-6_4)) contains supplementary material, which is available to authorized users.

where $\xi = \{\xi_j\}_{j \in \mathbb{N}}$ is an i.i.d. sequence, independent of v_0 , with $\xi_0 \sim N(0, \Sigma)$. The data is given by equation (2.2):

$$y_{j+1} = h(v_{j+1}) + \eta_{j+1}, \quad j \in \mathbb{Z}^+,$$

where $h : \mathbb{R}^n \rightarrow \mathbb{R}^m$ and $\eta = \{\eta_j\}_{j \in \mathbb{Z}^+}$ is an i.i.d. sequence, independent of (v_0, ξ) , with $\eta_1 \sim N(0, \Gamma)$.

4.1 Linear Gaussian Problems: The Kalman Filter

This algorithm provides a sequential method for updating the filtering distribution $\mathbb{P}(v_j|Y_j)$ from time j to time $j+1$, when Ψ and h are linear maps. In this case, the filtering distribution is Gaussian, and it can be characterized entirely through its mean and covariance. To see this, we note that the prediction step preserves Gaussianity by Lemma 1.5; the analysis step preserves Gaussianity because it is an application of Bayes's formula (1.7), and then Lemma 1.6 establishes the required Gaussian property, since the log pdf is quadratic in the unknown.

To be concrete, we let

$$\Psi(v) = Mv, \quad h(v) = Hv \tag{4.2}$$

for matrices $M \in \mathbb{R}^{n \times n}, H \in \mathbb{R}^{m \times n}$. We assume that $m \leq n$ and $\text{Rank}(H) = m$. We let (m_j, C_j) denote the mean and covariance of $v_j|Y_j$, noting that this entirely characterizes the random variable, since it is Gaussian. We let $(\hat{m}_{j+1}, \hat{C}_{j+1})$ denote the mean and covariance of $v_{j+1}|Y_j$, noting that this, too, completely characterizes the random variable, since it is also Gaussian. We now derive the map $(m_j, C_j) \mapsto (\hat{m}_{j+1}, \hat{C}_{j+1})$, using the intermediate variables $(\hat{m}_{j+1}, \hat{C}_{j+1})$, so that we may compute the prediction and analysis steps separately. This gives the Kalman filter in a form in which the update is expressed in terms of precision rather than covariance.

Theorem 4.1. *Assume that $C_0, \Gamma, \Sigma > 0$. Then $C_j > 0$ for all $j \in \mathbb{Z}^+$, and*

$$C_{j+1}^{-1} = (MC_jM^T + \Sigma)^{-1} + H^T\Gamma^{-1}H, \tag{4.3a}$$

$$C_{j+1}^{-1}m_{j+1} = (MC_jM^T + \Sigma)^{-1}Mm_j + H^T\Gamma^{-1}y_{j+1}. \tag{4.3b}$$

Proof We assume for the purposes of induction that $C_j > 0$, noting that this is true for $j = 0$ by assumption. The prediction step is determined by (2.1) in the case $\Psi(\cdot) = M\cdot$:

$$v_{j+1} = Mv_j + \xi_j, \quad \xi_j \sim N(0, \Sigma).$$

From this, it is clear that

$$\mathbb{E}(v_{j+1}|Y_j) = \mathbb{E}(Mv_j|Y_j) + \mathbb{E}(\xi_j|Y_j).$$

Since ξ_j is independent of Y_j , we have

$$\hat{m}_{j+1} = Mm_j. \tag{4.4}$$

Similarly,

$$\begin{aligned} \mathbb{E}((v_{j+1} - \hat{m}_{j+1}) \otimes (v_{j+1} - \hat{m}_{j+1})|Y_j) &= \mathbb{E}(M(v_j - m_j) \otimes M(v_j - m_j)|Y_j) + \mathbb{E}(\xi_j \otimes \xi_j|Y_j) \\ &\quad + \mathbb{E}(M(v_j - m_j) \otimes \xi_j|Y_j) + \mathbb{E}(\xi_j \otimes M(v_j - m_j)|Y_j). \end{aligned}$$

Again, since ξ_j is independent of Y_j and v_j , we have

$$\begin{aligned}\widehat{C}_{j+1} &= M\mathbb{E}((v_j - m_j) \otimes (v_j - m_j)|Y_j)M^T + \Sigma \\ &= MC_jM^T + \Sigma.\end{aligned}\quad (4.5)$$

Note that $\widehat{C}_{j+1} > 0$, because $C_j > 0$ by the inductive hypothesis and $\Sigma > 0$ by assumption.

Now we consider the analysis step. By (2.31), which is just Bayes's formula, and using Gaussianity, we have

$$\exp\left(-\frac{1}{2}|v - m_{j+1}|_{C_{j+1}}^2\right) \propto \exp\left(-\frac{1}{2}|\Gamma^{-\frac{1}{2}}(y_{j+1} - Hv)|^2 - \frac{1}{2}|\widehat{C}_{j+1}^{-\frac{1}{2}}(v - \widehat{m}_{j+1})|^2\right)\quad (4.6a)$$

$$= \exp\left(-\frac{1}{2}|y_{j+1} - Hv|_\Gamma^2 - \frac{1}{2}|v - \widehat{m}_{j+1}|_{\widehat{C}_{j+1}}^2\right).\quad (4.6b)$$

Equating quadratic terms in v gives, since $\Gamma > 0$ by assumption,

$$C_{j+1}^{-1} = \widehat{C}_{j+1}^{-1} + H^T \Gamma^{-1} H,\quad (4.7)$$

and equating linear terms in v gives¹

$$C_{j+1}^{-1} m_{j+1} = \widehat{C}_{j+1}^{-1} \widehat{m}_{j+1} + H^T \Gamma^{-1} y_{j+1}.\quad (4.8)$$

Substituting the expressions (4.4) and (4.5) for $(\widehat{m}_{j+1}, \widehat{C}_{j+1})$ gives the desired result. It remains to verify that $C_{j+1} > 0$. From (4.7), it follows, since $\Gamma^{-1} > 0$ by assumption and $\widehat{C}_{j+1} > 0$ (proved above), that $C_{j+1}^{-1} > 0$. Hence $C_{j+1} > 0$, and the induction is complete. \square

We may now reformulate the Kalman filter using covariances directly, rather than using precisions.

Corollary 4.2. *Under the assumptions of Theorem 4.1, the formulas for the Kalman filter given there may be rewritten as follows:*

$$\begin{aligned}d_{j+1} &= y_{j+1} - H\widehat{m}_{j+1}, \\ S_{j+1} &= H\widehat{C}_{j+1}H^T + \Gamma, \\ K_{j+1} &= \widehat{C}_{j+1}H^T S_{j+1}^{-1}, \\ m_{j+1} &= \widehat{m}_{j+1} + K_{j+1}d_{j+1}, \\ C_{j+1} &= (I - K_{j+1}H)\widehat{C}_{j+1},\end{aligned}$$

with $(\widehat{m}_{j+1}, \widehat{C}_{j+1})$ given in (4.4), (4.5).

Proof By (4.7), we have

$$C_{j+1}^{-1} = \widehat{C}_{j+1}^{-1} + H^T \Gamma^{-1} H,$$

and application of Lemma 4.4 below gives

$$\begin{aligned}C_{j+1} &= \widehat{C}_{j+1} - \widehat{C}_{j+1}H^T(\Gamma + H\widehat{C}_{j+1}H^T)^{-1}H\widehat{C}_{j+1} \\ &= (I - \widehat{C}_{j+1}H^T(\Gamma + H\widehat{C}_{j+1}H^T)^{-1}H)\widehat{C}_{j+1} \\ &= (I - \widehat{C}_{j+1}H^T S_{j+1}^{-1} H)\widehat{C}_{j+1} \\ &= (I - K_{j+1}H)\widehat{C}_{j+1},\end{aligned}$$

¹ We do not need to match the constant terms (with respect to v), since the normalization constant in Bayes's theorem deals with matching these.

as required. Then the identity (4.8) gives

$$\begin{aligned} m_{j+1} &= C_{j+1}\hat{C}_{j+1}^{-1}\hat{m}_{j+1} + C_{j+1}H^T\Gamma^{-1}y_{j+1} \\ &= (I - K_{j+1}H)\hat{m}_{j+1} + C_{j+1}H^T\Gamma^{-1}y_{j+1}. \end{aligned} \quad (4.9)$$

Now note that again by (4.7),

$$C_{j+1}(\hat{C}_{j+1}^{-1} + H^T\Gamma^{-1}H) = I,$$

so that

$$\begin{aligned} C_{j+1}H^T\Gamma^{-1}H &= I - C_{j+1}\hat{C}_{j+1}^{-1} \\ &= I - (I - K_{j+1}H) \\ &= K_{j+1}H. \end{aligned}$$

Since H has rank m , we deduce that

$$C_{j+1}H^T\Gamma^{-1} = K_{j+1}.$$

Hence (4.9) gives

$$m_{j+1} = (I - K_{j+1}H)\hat{m}_{j+1} + K_{j+1}y_{j+1} = \hat{m}_{j+1} + K_{j+1}d_{j+1},$$

as required. \square

Remark 4.3. The key difference between the Kalman update formulas in Theorem 4.1 and those in Corollary 4.2 is that in the former, matrix inversion takes place in the state space, with dimension n , while in the latter matrix, inversion takes place in the data space, with dimension m . In many applications, $m \ll n$, since the observed subspace dimension is much less than the state space dimension, and thus the formulation in Corollary 4.2 is frequently employed in practice. The quantity d_{j+1} is referred to as the innovation at time step $j+1$. It measures the mismatch of the predicted state from the data. The matrix K_{j+1} is known as the Kalman gain. ♠

The following matrix identity was used to derive the formulation of the Kalman filter in which inversion takes place in the data space.

Lemma 4.4. Woodbury Matrix Identity Let $A \in \mathbb{R}^{p \times p}$, $U \in \mathbb{R}^{p \times q}$, $C \in \mathbb{R}^{q \times q}$, and $V \in \mathbb{R}^{q \times p}$. If A and C are positive, then $A + UCV$ is invertible, and

$$(A + UCV)^{-1} = A^{-1} - A^{-1}U\left(C^{-1} + VA^{-1}U\right)^{-1}VA^{-1}.$$

4.2 Approximate Gaussian Filters

Here we introduce a family of methods, based on invoking a minimization principle that underlies the Kalman filter, that has a natural generalization to non-Gaussian problems. The update equation for the Kalman filter mean, (4.8), can be written as

$$m_{j+1} = \arg \min_v \mathcal{I}_{\text{filter}}(v),$$

where

$$\mathbf{I}_{\text{filter}}(v) := \frac{1}{2}|y_{j+1} - Hv|_F^2 + \frac{1}{2}|v - \hat{m}_{j+1}|_{\hat{C}_{j+1}}^2; \quad (4.10)$$

here \hat{m}_{j+1} is calculated from (4.4), and \hat{C}_{j+1} is given by (4.5). The fact that this minimization principle holds follows from (4.6). (We note that $\mathbf{I}_{\text{filter}}(\cdot)$ in fact depends on j , but we suppress explicit reference to this dependence for notational simplicity.)

While the Kalman filter itself is restricted to linear Gaussian problems, the formulation via minimization generalizes to nonlinear problems. A natural generalization of (4.10) to the nonlinear case is to define

$$\mathbf{I}_{\text{filter}}(v) := \frac{1}{2}|y_{j+1} - h(v)|_F^2 + \frac{1}{2}|v - \hat{m}_{j+1}|_{\hat{C}_{j+1}}^2, \quad (4.11)$$

where

$$\hat{m}_{j+1} = \Psi(m_j) + \xi_j,$$

and then to set

$$m_{j+1} = \arg \min_v \mathbf{I}_{\text{filter}}(v).$$

This provides a family of algorithms for updating the mean that depend on how \hat{C}_{j+1} is specified. In this section, we will consider several choices for this specification, and hence several different algorithms. Notice that the minimization principle is very natural: it enforces a compromise between fitting the model prediction \hat{m}_{j+1} and the data y_{j+1} .

For simplicity, we consider the case in which observations are linear and $h(v) = Hv$, leading to the update algorithm $m_j \mapsto m_{j+1}$ defined by

$$\hat{m}_{j+1} = \Psi(m_j) + \xi_j, \quad (4.12a)$$

$$\mathbf{I}_{\text{filter}}(v) = \frac{1}{2}|y_{j+1} - Hv|_F^2 + \frac{1}{2}|v - \hat{m}_{j+1}|_{\hat{C}_{j+1}}^2, \quad (4.12b)$$

$$m_{j+1} = \arg \min_v \mathbf{I}_{\text{filter}}(v). \quad (4.12c)$$

This quadratic minimization problem is explicitly solvable, and by the arguments used in deriving Corollary 4.2, we deduce the following update formulas:

$$m_{j+1} = (I - K_{j+1}H)\hat{m}_{j+1} + K_{j+1}y_{j+1}, \quad (4.13a)$$

$$K_{j+1} = \hat{C}_{j+1}H^T S_{j+1}^{-1}, \quad (4.13b)$$

$$S_{j+1} = H\hat{C}_{j+1}H^T + \Gamma. \quad (4.13c)$$

The next three subsections correspond to algorithms derived in this way, namely by minimizing $\mathbf{I}_{\text{filter}}(v)$, but corresponding to different choices of the model covariance \hat{C}_{j+1} . We also note that in the first two of these subsections, we choose $\xi_j \equiv 0$ in equation (4.12a), so that the prediction is made by the noise-free dynamical model; however, that is not a necessary choice, and while it is natural for the extended Kalman filter for 3DVAR, including random effects in (4.12a) is also reasonable in some settings. Likewise, the ensemble Kalman filter can also be implemented with noise-free prediction models.

We refer to these three algorithms collectively as **approximate Gaussian filters**. This is because they invoke a Gaussian approximation when they update the estimate of the signal via (4.12b). Specifically, this update is the correct update for the mean if the assumption that $\mathbb{P}(v_{j+1}|Y_j) = N(\hat{m}_{j+1}), \hat{C}_{j+1})$ is invoked for the prediction step. In general, the approximation implied by this assumption will not be a good one, and this can invalidate the statistical

accuracy of the resulting algorithms. However, the resulting algorithms may still have desirable properties in terms of signal estimation; in Section 4.4.2, we will demonstrate that this is indeed so.

4.2.1. 3DVAR

This algorithm is derived from (4.13) by simply fixing the model covariance $\hat{C}_{j+1} \equiv \hat{C}$ for all j . Thus we obtain

$$\hat{m}_{j+1} = \Psi(m_j), \quad (4.14a)$$

$$m_{j+1} = (I - KH)\hat{m}_{j+1} + Ky_{j+1}, \quad (4.14b)$$

$$K = \hat{C}H^T S^{-1}, \quad S = H\hat{C}H^T + \Gamma. \quad (4.14c)$$

The nomenclature 3DVAR refers to the fact that the method is variational (it is based on the minimization principle underlying all of the approximate Gaussian methods), and it works sequentially at each fixed time j ; as such, the minimization, when applied to practical physical problems, is over three spatial dimensions. This should be contrasted with 4DVAR, which involves a minimization over all spatial dimensions, as well as time—four dimensions in all.

We now describe two methodologies that generalize 3DVAR by employing model covariances that evolve from step j to step $j + 1$: the extended and ensemble Kalman filters. We present both methods in basic form but conclude the section with some discussion of methods widely used in practice to improve their practical performance.

4.2.2. Extended Kalman Filter

The idea of the extended Kalman filter (ExKF) is to propagate covariances according to the linearization of (2.1) and to propagate the mean using (2.3). Thus we obtain, from modification of Corollary 4.2 and (4.4), (4.5),

$$\begin{aligned} \text{Prediction} & \left\{ \begin{array}{l} \hat{m}_{j+1} = \Psi(m_j), \\ \hat{C}_{j+1} = D\Psi(m_j)C_jD\Psi(m_j)^T + \Sigma. \end{array} \right. \\ \text{Analysis} & \left\{ \begin{array}{l} S_{j+1} = H\hat{C}_{j+1}H^T + \Gamma, \\ K_{j+1} = \hat{C}_{j+1}H^T S_{j+1}^{-1}, \\ m_{j+1} = (I - K_{j+1}H)\hat{m}_{j+1} + K_{j+1}y_{j+1}, \\ C_{j+1} = (I - K_{j+1}H)\hat{C}_{j+1}. \end{array} \right. \end{aligned}$$

4.2.3. Ensemble Kalman Filter

The ensemble Kalman filter (EnKF) generalizes the idea of approximate Gaussian filters in a significant way: rather than using the minimization procedure (4.12) to update a single estimate of the *mean*, it is used to generate an *ensemble* of particles all of which satisfy the

model/data compromise inherent in the minimization; the mean and covariance used in the minimization are then estimated using this ensemble, thereby adding further coupling to the particles, in addition to that introduced by the data.

The EnKF is executed in a variety of ways, and we begin by describing one of these, the perturbed observation EnKF:

$$\text{Prediction} \quad \begin{cases} \hat{v}_{j+1}^{(n)} = \Psi(v_j^{(n)}) + \xi_j^{(n)}, & n = 1, \dots, N, \\ \hat{m}_{j+1} = \frac{1}{N} \sum_{n=1}^N \hat{v}_{j+1}^{(n)}, \\ \hat{C}_{j+1} = \frac{1}{N-1} \sum_{n=1}^N (\hat{v}_{j+1}^{(n)} - \hat{m}_{j+1})(\hat{v}_{j+1}^{(n)} - \hat{m}_{j+1})^T. \end{cases}$$

$$\text{Analysis} \quad \begin{cases} S_{j+1} = H\hat{C}_{j+1}H^T + \Gamma, \\ K_{j+1} = \hat{C}_{j+1}H^T S_{j+1}^{-1}, \\ v_{j+1}^{(n)} = (I - K_{j+1}H)\hat{v}_{j+1}^{(n)} + K_{j+1}y_{j+1}^{(n)}, & n = 1, \dots, N, \\ y_{j+1}^{(n)} = y_{j+1} + \eta_{j+1}^{(n)}, & n = 1, \dots, N. \end{cases}$$

Here $\eta_j^{(n)}$ are i.i.d. draws from $N(0, \Gamma)$, and $\xi_j^{(n)}$ are i.i.d. draws from $N(0, \Sigma)$. ‘‘Perturbed observation’’ refers to the fact that each particle sees an observation perturbed by an independent draw from $N(0, \Gamma)$. This procedure gives the Kalman filter in the linear case in the limit of an infinite ensemble. Even though the algorithm is motivated through our general approximate Gaussian filters framework, notice that the ensemble is not prescribed to be Gaussian. Indeed, it evolves under the full nonlinear dynamics in the prediction step. This fact, together with the fact that covariance matrices are not propagated explicitly, other than through the empirical properties of the ensemble, has made the algorithm very appealing to practitioners.

Another way to motivate the preceding algorithm is to introduce the family of cost functions

$$l_{\text{filter},n}(v) := \frac{1}{2}|y_{j+1}^{(n)} - Hv|_\Gamma^2 + \frac{1}{2}|v - \hat{v}_{j+1}^{(n)}|_{\hat{C}_{j+1}}^2. \quad (4.15)$$

The analysis step proceeds to determine the ensemble $\{v_{j+1}^{(n)}\}_{n=1}^N$ by minimizing $l_{\text{filter},n}$ with $n = 1, \dots, N$. The set $\{\hat{v}_{j+1}^{(n)}\}_{n=1}^N$ is found from running the prediction step using the fully nonlinear dynamics. These minimization problems are coupled through \hat{C}_{j+1} , which depends on the entire set of $\{\hat{v}_j^{(n)}\}_{n=1}^N$. The algorithm thus provides update rules of the form

$$\{v_j^{(n)}\}_{n=1}^N \mapsto \{\hat{v}_{j+1}^{(n)}\}_{n=1}^N, \quad \{\hat{v}_{j+1}^{(n)}\}_{n=1}^N \mapsto \{v_{j+1}^{(n)}\}_{n=1}^N, \quad (4.16)$$

defining approximations of the prediction and analysis steps respectively.

It is then natural to think of the algorithm making the approximations

$$\mu_j \approx \mu_j^N = \frac{1}{N} \sum_{n=1}^N \delta_{v_j^{(n)}}, \quad \hat{\mu}_{j+1} \approx \hat{\mu}_{j+1}^N = \frac{1}{N} \sum_{n=1}^N \delta_{\hat{v}_{j+1}^{(n)}}. \quad (4.17)$$

Thus we have a form of Monte Carlo approximation of the distribution of interest. However, except for linear problems, the approximations given do not, in general, converge to the true distributions μ_j and $\hat{\mu}_j$ as $N \rightarrow \infty$.

4.2.4. Ensemble Square-Root Kalman Filter

We now describe another popular variant of the EnKF. The idea of this variant is to define the analysis step in such a way that an ensemble of particles is produced whose empirical covariance *exactly* satisfies the Kalman identity

$$C_{j+1} = (I - K_{j+1}H)\widehat{C}_{j+1}, \quad (4.18)$$

which relates the covariances in the analysis step to those in the prediction step. This is done by mapping the mean of the predicted ensemble according to the standard Kalman update and introducing a linear deterministic transformation of the differences between the particle positions and their mean to enforce (4.18). Doing so eliminates a sampling error inherent in the perturbed observation approach. The resulting algorithm has the following form:

$$\begin{aligned} \text{Prediction} & \left\{ \begin{array}{l} \widehat{v}_{j+1}^{(n)} = \Psi(v_j^{(n)}) + \xi_j^{(n)}, \quad n = 1, \dots, N, \\ \widehat{m}_{j+1} = \frac{1}{N} \sum_{n=1}^N \widehat{v}_{j+1}^{(n)}, \\ \widehat{C}_{j+1} = \frac{1}{N-1} \sum_{n=1}^N (\widehat{v}_{j+1}^{(n)} - \widehat{m}_{j+1})(\widehat{v}_{j+1}^{(n)} - \widehat{m}_{j+1})^T, \end{array} \right. \\ \text{Analysis} & \left\{ \begin{array}{l} S_{j+1} = H\widehat{C}_{j+1}H^T + \Gamma, \\ K_{j+1} = \widehat{C}_{j+1}H^T S_{j+1}^{-1}, \\ m_{j+1} = (I - K_{j+1}H)\widehat{m}_{j+1} + K_{j+1}y_{j+1}, \\ v_{j+1}^{(n)} = m_{j+1} + \zeta_{j+1}^{(n)}. \end{array} \right. \end{aligned}$$

Here the $\{\zeta_{j+1}^{(n)}\}_{n=1}^N$ are designed to have sample covariance $C_{j+1} = (I - K_{j+1}H)\widehat{C}_{j+1}$. There are several ways to do this, and we now describe one of them, referred to as the ensemble transform Kalman filter (ETKF).

If we define

$$\widehat{X}_{j+1} = \frac{1}{\sqrt{N-1}} [\widehat{v}_{j+1}^{(1)} - \widehat{m}_{j+1}, \dots, \widehat{v}_{j+1}^{(N)} - \widehat{m}_{j+1}],$$

then $\widehat{C}_{j+1} = \widehat{X}_{j+1}\widehat{X}_{j+1}^T$. We now seek a transformation T_{j+1} such that if $X_{j+1} = \widehat{X}_{j+1}T_{j+1}^{\frac{1}{2}}$, then

$$C_{j+1} := X_{j+1}X_{j+1}^T = (I - K_{j+1}H)\widehat{C}_{j+1}. \quad (4.19)$$

Note that the X_{j+1} (respectively the \widehat{X}_{j+1}) correspond to Cholesky factors of the matrices C_{j+1} (respectively \widehat{C}_{j+1}) respectively. We may now define the $\{\zeta_{j+1}^{(n)}\}_{n=1}^N$ by

$$X_{j+1} = \frac{1}{\sqrt{N-1}} [\zeta_{j+1}^{(1)}, \dots, \zeta_{j+1}^{(N)}].$$

We now demonstrate how to find an appropriate transformation T_{j+1} . We assume that T_{j+1} is symmetric and positive definite and that the standard matrix square root is employed. Choosing

$$T_{j+1} = [I + (H\widehat{X}_{j+1})^T \Gamma^{-1} (H\widehat{X}_{j+1})]^{-1},$$

we see that

$$\begin{aligned}
X_{j+1} X_{j+1}^T &= \hat{X}_{j+1} T_{j+1} \hat{X}_{j+1}^T \\
&= \hat{X}_{j+1} \left[I + (H \hat{X}_{j+1})^T \Gamma^{-1} (H \hat{X}_{j+1}) \right]^{-1} \hat{X}_{j+1}^T \\
&= \hat{X}_{j+1} \left\{ I - (H \hat{X}_{j+1})^T \left[(H \hat{X}_{j+1})(H \hat{X}_{j+1})^T + \Gamma \right]^{-1} (H \hat{X}_{j+1}) \right\} \hat{X}_{j+1}^T \\
&= (I - K_{j+1} H) \hat{C}_{j+1},
\end{aligned}$$

as required, where the transformation between the second and third lines is justified by Lemma 4.4. It is important to ensure that $\mathbf{1}$, the vector of all ones, is an eigenvector of the transformation T_{j+1} , and hence of $T_{j+1}^{\frac{1}{2}}$, so that the mean of the ensemble is preserved. This is guaranteed by T_{j+1} as defined.

4.3 The Particle Filter

In this section, we introduce an important class of filtering methods known as *particle filters*. In contrast to the filters introduced in the preceding section, the particle filter can be *proved* to reproduce the true posterior filtering distribution in the large-particle limit, and as such, has a privileged place among all the filters introduced in this book. We will describe the method in its basic form—the *bootstrap filter*—and then give a proof of convergence. It is important to appreciate that the form of particle filter introduced here is far from state-of-the-art, and that far more sophisticated versions are used in practical applications. Nonetheless, despite this sophistication, particle filters do not perform well in applications such as those arising in geophysical applications of data assimilation, because the data in those applications places very strong constraints on particle locations, making efficient algorithms very hard to design. It is for this reason that we have introduced particle filters after the approximate Gaussian filters introduced in the preceding section. The filters in the preceding section tend to be more robust to data specifications. However, they all rely on the invocation of ad hoc Gaussian assumptions in their derivation and hence do not provably produce the correct posterior filtering distribution, notwithstanding their ability, in partially observed small-noise scenarios, to correctly identify the signal itself, as in Theorem 4.10. Because it can provably reproduce the correct filtering distribution, the particle filter thus plays an important role, conceptually, even though it is not, in current form, a practical algorithm in geophysical applications. With further improvements it may, in time, form the basis for practical algorithms in geophysical applications.

4.3.1. The Basic Approximation Scheme

All probability measures that possess density with respect to Lebesgue measure can be approximated by a finite convex combination of Dirac probability measures; an example of this is the **Monte Carlo sampling** idea that we described at the start of Chapter 3, and it also underlies the ensemble Kalman filter of Section 4.2.3. In practice, the idea of approximation by a convex combination of probability measures requires the determination of the locations and weights associated with these Dirac measures. Particle filters are sequential algorithms that use this idea to approximate the true filtering distribution $\mathbb{P}(v_j | Y_j)$.

Basic Monte Carlo, as in (3.1), and the ensemble Kalman filter, as in (4.17), correspond to approximation by equal weights. Recall μ_j , the probability measure on \mathbb{R}^n corresponding to

the density $\mathbb{P}(v_j|Y_j)$, and $\hat{\mu}_{j+1}$, the probability measure on \mathbb{R}^n corresponding to the density $\mathbb{P}(v_{j+1}|Y_j)$. The basic form of the particle filter proceeds by allowing the weights to vary and by finding N -particle Dirac measure approximations of the form

$$\mu_j \approx \mu_j^N := \sum_{n=1}^N w_j^{(n)} \delta_{v_j^{(n)}}, \quad \hat{\mu}_{j+1} \approx \hat{\mu}_{j+1}^N := \sum_{n=1}^N \hat{w}_{j+1}^{(n)} \delta_{\hat{v}_{j+1}^{(n)}}. \quad (4.20)$$

The weights must sum to 1. The approximate distribution μ_j^N is completely defined by particle positions $v_j^{(n)}$ and weights $w_j^{(n)}$, and the approximate distribution $\hat{\mu}_{j+1}^N$ is completely defined by particle positions $\hat{v}_{j+1}^{(n)}$ and weights $\hat{w}_{j+1}^{(n)}$. Thus the objective of the method is to find update rules

$$\{v_j^{(n)}, w_j^{(n)}\}_{n=1}^N \mapsto \{\hat{v}_{j+1}^{(n)}, \hat{w}_{j+1}^{(n)}\}_{n=1}^N, \quad \{\hat{v}_{j+1}^{(n)}, \hat{w}_{j+1}^{(n)}\}_{n=1}^N \mapsto \{v_{j+1}^{(n)}, w_{j+1}^{(n)}\}_{n=1}^N \quad (4.21)$$

defining the prediction and analysis approximations respectively; compare this with (4.16) for the EnKF, where the particle weights are uniform and only the positions are updated. Defining the updates for the particle filter may be achieved by an application of sampling for the prediction step, and of Bayesian probability for the analysis step.

Recall the prediction and analysis formulas from (2.30) and (2.31), which can be summarized as

$$\mathbb{P}(v_{j+1}|Y_j) = \int_{\mathbb{R}^n} \mathbb{P}(v_{j+1}|v_j) \mathbb{P}(v_j|Y_j) dv_j, \quad (4.22a)$$

$$\mathbb{P}(v_{j+1}|Y_{j+1}) = \frac{\mathbb{P}(y_{j+1}|v_{j+1}) \mathbb{P}(v_{j+1}|Y_j)}{\mathbb{P}(y_{j+1}|Y_j)}. \quad (4.22b)$$

We may rewrite (4.22) as

$$\hat{\mu}_{j+1}(\cdot) = (P\mu_j)(\cdot) := \int_{\mathbb{R}^n} \mathbb{P}(\cdot|v_j) \mu_j(dv_j) \quad (4.23a)$$

$$\frac{d\mu_{j+1}}{d\hat{\mu}_{j+1}}(v_{j+1}) = \frac{\mathbb{P}(y_{j+1}|v_{j+1})}{\mathbb{P}(y_{j+1}|Y_j)}. \quad (4.23b)$$

Writing the update formulas in this way is important for us, because they then make sense in the absence of Lebesgue densities; in particular, we can use them in situations where Dirac masses appear, as they do in our approximate probability measures. The formula (4.23b) for the *density* or *Radon–Nikodym derivative* of μ_{j+1} with respect to that of $\hat{\mu}_{j+1}$ has a straightforward interpretation: the right-hand side quantifies how to reweight expectations under $\hat{\mu}_{j+1}$ so that they become expectations under μ_{j+1} . To be concrete, we may write

$$\mathbb{E}^{\mu_{j+1}} \varphi(v_{j+1}) = \mathbb{E}^{\hat{\mu}_{j+1}} \left(\frac{d\mu_{j+1}}{d\hat{\mu}_{j+1}}(v_{j+1}) \varphi(v_{j+1}) \right).$$

4.3.2. Sequential Importance Resampling

The simplest particle filter, which is based on *sequential importance resampling*, is now described. We begin by assuming that we have an approximation μ_j^N given by (4.20) and explain

how to evolve the weights $\{v_j^{(n)}, w_j^{(n)}\}_{n=1}^N$ into $\{v_j^{(n+1)}, w_j^{(n+1)}\}_{n=1}^N$, via $\{\hat{v}_{j+1}^{(n)}, \hat{w}_{j+1}^{(n)}\}_{n=1}^N$, as in (4.21).

Prediction. In this step, we approximate the prediction phase of the Markov chain. To do this, we simply draw $\hat{v}_{j+1}^{(n)}$ from the kernel p of the Markov chain (2.1a) started from $v_j^{(n)}$. Thus the relevant kernel is $p(v_j, v_{j+1}) = \mathbb{P}(v_{j+1}|v_j)$. We then have $\hat{v}_{j+1}^{(n)} \sim p(v_j^{(n)}, \cdot)$. We leave the weights of the approximation unchanged, so that $\hat{w}_{j+1}^{(n)} = w_j^{(n)}$. From these new particles and (in fact unchanged) weights, we have the particle approximation

$$\hat{\mu}_{j+1}^N = \sum_{n=1}^N w_j^{(n)} \delta_{\hat{v}_{j+1}^{(n)}}. \quad (4.24)$$

Analysis. In this step, we approximate the incorporation of data via Bayes's formula. Define $g_j(v)$ by

$$g_j(v_{j+1}) \propto \mathbb{P}(y_{j+1}|v_{j+1}), \quad (4.25)$$

where the constant of proportionality is, for example, the normalization for the Gaussian and is hence independent of both y_{j+1} and v_{j+1} . We now apply Bayes's formula in the form (4.23b). Thus we obtain

$$\mu_{j+1}^N = \sum_{n=1}^N w_{j+1}^{(n)} \delta_{\hat{v}_{j+1}^{(n)}}, \quad (4.26)$$

where

$$w_{j+1}^{(n)} = \tilde{w}_{j+1}^{(n)} / \left(\sum_{n=1}^N \tilde{w}_{j+1}^{(n)} \right), \quad \tilde{w}_{j+1}^{(n)} = g_j(\hat{v}_{j+1}^{(n)}) w_j^{(n)}. \quad (4.27)$$

The first equation in the preceding is required for normalization. Thus in this step, we do not change the particle positions, but we reweight them.

Resampling. The algorithm as described is deficient in two respects, both of which can be dealt with by introducing a resampling step into the algorithm. Firstly, the initial measure μ_0 for the true filtering distribution will not typically be made up of a combination of Dirac measures. Secondly, the method can perform poorly if one of the particle weights approaches 1 (and then all others approach 0). The effect of the first can be dealt with by sampling the initial measure and approximating it by an equally weighted (by N^{-1}) sum of Dirac measures at the samples. The second can be ameliorated by drawing a set of N particles from the measure (4.26) and assigning weight N^{-1} to each; this has the effect of multiplying particles with high weights and killing particles with low weights.

Putting together the three preceding steps leads to the following algorithm; for notational convenience, we use Y_0 to denote the empty vector (no observations at the start):

1. Set $j = 0$ and $\mu_0^N(dv_0) = \mu_0(dv_0)$.
2. Draw $v_j^{(n)} \sim \mu_j^N$, $n = 1, \dots, N$.
3. Set $w_j^{(n)} = 1/N$, $n = 1, \dots, N$; redefine $\mu_j^N := \sum_{n=1}^N w_j^{(n)} \delta_{v_j^{(n)}}$.
4. Draw $\hat{v}_{j+1}^{(n)} \sim p(v_j^{(n)} | \cdot)$.
5. Define $w_{j+1}^{(n)}$ by (4.27) and $\mu_{j+1}^N := \sum_{n=1}^N w_{j+1}^{(n)} \delta_{\hat{v}_{j+1}^{(n)}}$.
6. $j + 1 \rightarrow j$.
7. Go to step 2.

This algorithm is conceptually intuitive, proposing that each particle moves according to the dynamics of the underlying model itself, and is then reweighted according to the likelihood of the proposed particle, i.e., according to the data. This sequential importance resampling filter is also sometimes termed the **bootstrap filter**. We will comment on important improvements to this basic algorithm in the following section and in the bibliographic notes. Here we prove convergence of this basic method, as the number of particles goes to infinity, thereby demonstrating the potential power of the bootstrap filter and more sophisticated variants of it.

Recall that by (2.32), the true filtering distribution simply satisfies the iteration

$$\mu_{j+1} = L_j P \mu_j, \quad \mu_0 = N(m_0, C_0), \quad (4.28)$$

where P corresponds to moving a point currently at v according to the Markov kernel $p(\cdot|v)$ describing the dynamics given by (2.1a), and L_j denotes the application of Bayes's formula with likelihood proportional to $g_j(\cdot)$ given by (4.25). Recall also the sampling operator S^N defined by (3.1). It is then instructive to write the particle filtering algorithm that approximates (4.28) in the following form:

$$\mu_{j+1}^N = L_j S^N P \mu_j^N, \quad \mu_0^N = \mu_0. \quad (4.29)$$

There is a slight trickery here in writing application of the sampling S^N *after* application of P , but some reflection shows that this is well justified: applying P followed by S^N can be shown, by first conditioning on the initial point and sampling with respect to P , and then sampling over the distribution of the initial point, to be the algorithm as defined.

Comparison of (4.28) and (4.29) shows that analyzing the particle filter requires estimation of the error induced by application of S^N (the *resampling error*) together with estimation of the rate of accumulation of this error in time under the application of L_j and P . We now build the tools to allow us to do this. The operators L_j , P , and S^N map the space $\mathcal{P}(\mathbb{R}^n)$ of probability measures on \mathbb{R}^n into itself according to the following:

$$(L_j \mu)(dv) = \frac{g_j(v)\mu(dv)}{\int_{\mathbb{R}^n} g_j(v)\mu(dv)}, \quad (4.30a)$$

$$(P\mu)(dv) = \int_{\mathbb{R}^n} p(v', dv)\mu(dv'), \quad (4.30b)$$

$$(S^N \mu)(dv) = \frac{1}{N} \sum_{n=1}^N \delta_{v^{(n)}}(dv), \quad v^{(n)} \sim \mu \quad \text{i.i.d.} \quad (4.30c)$$

Notice that both L_j and P are deterministic maps, while S^N is random. Let $\mu = \mu_\omega$ denote, for each ω , an element of $\mathcal{P}(\mathbb{R}^n)$. If we then assume that ω is a random variable describing the randomness required to define the sampling operator S^N , and let \mathbb{E}^ω denote expectation over ω , then we may define a “root mean square” distance $d(\cdot, \cdot)$ between two random probability measures μ_ω, ν_ω as follows:

$$d(\mu, \nu) = \sup_{|f|_\infty \leq 1} \sqrt{\mathbb{E}^\omega |\mu(f) - \nu(f)|^2}.$$

Here we have used the convention that $\mu(f) = \int_{\mathbb{R}^n} f(v)\mu(dv)$ for measurable $f : \mathbb{R}^n \rightarrow \mathbb{R}$, and similar for ν . Furthermore,

$$|f|_\infty = \sup_u |f(u)|.$$

This distance indeed generates a metric, and in particular, satisfies the triangle inequality. Note also that in the absence of randomness within the measures, the metric satisfies $d(\mu, \nu) = 2d_{\text{TV}}(\mu, \nu)$, by (1.12); that is, it reduces to the total variation metric. In our context, the randomness within the probability measures comes from the sampling operator S^N used to define the numerical approximation.

Theorem 4.5. *We assume in the following that there exists $\kappa \in (0, 1]$ such that for all $v \in \mathbb{R}^n$ and $j \in \mathbb{N}$,*

$$\kappa \leq g_j(v) \leq \kappa^{-1}.$$

Then

$$d(\mu_J^N, \mu_J) \leq \sum_{j=1}^J (2\kappa^{-2})^j \frac{1}{\sqrt{N}}.$$

Proof The desired result is proved below in a straightforward way from the following three facts, whose proof we postpone to three lemmas at the end of the section:

$$\sup_{\mu \in \mathcal{P}(\mathbb{R}^n)} d(S^N \mu, \mu) \leq \frac{1}{\sqrt{N}}, \quad (4.31a)$$

$$d(P\nu, P\mu) \leq d(\nu, \mu), \quad (4.31b)$$

$$d(L_j \nu, L_j \mu) \leq 2\kappa^{-2} d(\nu, \mu). \quad (4.31c)$$

By the triangle inequality, we have, for $\nu_j^N = P\mu_j^N$,

$$\begin{aligned} d(\mu_{j+1}^N, \mu_{j+1}) &= d(L_j S^N P\mu_j^N, L_j P\mu_j) \\ &\leq d(L_j P\mu_j^N, L_j P\mu_j) + d(L_j S^N P\mu_j^N, L_j P\mu_j^N) \\ &\leq 2\kappa^{-2} \left(d(\mu_j^N, \mu_j) + d(S^N \nu_j^N, \nu_j^N) \right) \\ &\leq 2\kappa^{-2} \left(d(\mu_j^N, \mu_j) + \frac{1}{\sqrt{N}} \right). \end{aligned}$$

Iterating, after noting that $\mu_0^N = \mu_0$, gives the desired result. \square

Remark 4.6. *This important theorem shows that the particle filter reproduces the true filtering distribution in the large-particle limit. We make some comments about this.*

- This theorem shows that at every fixed discrete time j , the filtering distribution μ_j is well approximated by the bootstrap filtering distribution μ_j^N in the sense that as the number of particles N goes to ∞ , the approximating measure converges to the true measure. However, since $\kappa < 1$, the number of particles required to decrease the upper bound on the error beneath a specified tolerance grows with J .
- If the likelihoods have a small lower bound, then the constant in the convergence proof may be prohibitively expensive, requiring an enormous number of particles to obtain a small error. This is similar to the discussion concerning the independence dynamics sampler in Section 3.4, where we showed that large values in the potential Φ lead to slow convergence of the Markov chain, and the resultant need for a large number of samples.
- In fact, in many applications, the likelihoods g_j may not be bounded from above or below, uniformly in j , and more refined analysis is required. However, if the Markov kernel P is ergodic, then it is possible to obtain bounds in which the error constant arising in the analysis has milder growth with respect to J .

- Considering the case of deterministic dynamics shows just how difficult it may be to make the theorem applicable in practice: if the dynamics is deterministic, then the original set of samples from μ_0 , $\{v_0^{(n)}\}_{n=1}^N$, give rise to a set of particles $v_j^{(n)} = \Psi^{(j)}(v_0^{(n)})$; in other words, the particle positions are unaffected by the data. This is clearly a highly undesirable situation in general, since there is no reason at all why the pushforward under the dynamics of the initial measure μ_0 should have substantial overlap with the filtering distribution for a given fixed data set. Indeed, for chaotic dynamical systems, one would expect that it does not have such overlap, since the pushforward measure will be spread over the global attractor, while the data will, at fixed time, correspond to a single point on the attractor. This example motivates the improved proposals of the next section.



Before describing improvements to the basic particle filter, we prove the three lemmas underlying the convergence proof.

Lemma 4.7. *The sampling operator satisfies*

$$\sup_{\mu \in \mathcal{P}(\mathbb{R}^n)} d(S^N \mu, \mu) \leq \frac{1}{\sqrt{N}}.$$

Proof Let ν be an element of $\mathcal{P}(\mathbb{R}^n)$ and $\{v^{(n)}\}_{n=1}^N$ i.i.d. with $v^{(1)} \sim \nu$. In this proof, the randomness in the measure S^N arises from these samples $\{v^{(n)}\}_{n=1}^N$, and expectation over this randomness is denoted by \mathbb{E} . Then

$$S^N \nu(f) = \frac{1}{N} \sum_{n=1}^N f(v^{(n)}),$$

and defining $\bar{f} = f - \nu(f)$, we deduce that

$$S^N \nu(f) - \nu(f) = \frac{1}{N} \sum_{n=1}^N \bar{f}(v^{(n)}).$$

It is straightforward to see that

$$\mathbb{E} \bar{f}(v^{(n)}) \bar{f}(v^{(l)}) = \delta_{nl} \mathbb{E} |\bar{f}(v^{(n)})|^2.$$

Furthermore, for $|f|_\infty \leq 1$,

$$\mathbb{E} |\bar{f}(v^{(1)})|^2 = \mathbb{E} |f(v^{(1)})|^2 - |\mathbb{E} f(v^{(1)})|^2 \leq 1.$$

It follows that for $|f|_\infty \leq 1$,

$$\mathbb{E} |\nu(f) - S^N \nu(f)|^2 = \frac{1}{N^2} \sum_{n=1}^N \mathbb{E} |\bar{f}(v^{(n)})|^2 \leq \frac{1}{N}.$$

Since the result is independent of ν , we may take the supremum over all probability measures and obtain the desired result. \square

Lemma 4.8. *Since P is a Markov kernel, we have*

$$d(P\nu, P\nu') \leq d(\nu, \nu').$$

Proof Define

$$q(v') = \int_{\mathbb{R}^n} p(v', v) f(v) dv = \mathbb{E}(v_1 | v_0 = v'),$$

that is, the expected value of f under one step of the Markov chain given by (2.1a), started from v' . Clearly, for $|f|_\infty \leq 1$,

$$|q(v')| \leq \int_{\mathbb{R}^n} p(v', dv) |f(v)| \leq \int_{\mathbb{R}^n} p(v', dv) = 1.$$

Thus

$$|q|_\infty \leq \sup_v |q(v)| \leq 1.$$

Note that

$$\begin{aligned} \nu(q) &= \mathbb{E}(v_1 | v_0 \sim \nu) = \int_{\mathbb{R}^n} \int_{\mathbb{R}^n} p(v', v) f(v) \nu(dv') dv \\ &= \int_{\mathbb{R}^n} \left(\int_{\mathbb{R}^n} p(v', v) \nu(dv') \right) f(v) dv = P\nu(f). \end{aligned}$$

Thus $P\nu(f) = \nu(q)$, and it follows that

$$|P\nu(f) - P\nu'(f)| = |\nu(q) - \nu'(q)|.$$

Thus

$$\begin{aligned} d(P\nu, P\nu') &= \sup_{|f|_\infty \leq 1} \left(\mathbb{E}^\omega |P\nu(f) - P\nu'(f)|^2 \right)^{\frac{1}{2}} \\ &\leq \sup_{|q|_\infty \leq 1} \left(\mathbb{E}^\omega |\nu(q) - \nu'(q)|^2 \right)^{\frac{1}{2}} \\ &= d(\nu, \nu'), \end{aligned}$$

as required. \square

Lemma 4.9. *Under the assumptions of Theorem 4.5, we have*

$$d(L_j \nu, L_j \mu) \leq 2\kappa^{-2} d(\nu, \mu).$$

Proof Notice that for $|f|_\infty < \infty$, we can rewrite

$$(L_j \nu)(f) - (L_j \mu)(f) = \frac{\nu(fg_j)}{\nu(g_j)} - \frac{\mu(fg_j)}{\mu(g_j)} \quad (4.32a)$$

$$= \frac{\nu(fg_j)}{\nu(g_j)} - \frac{\mu(fg_j)}{\nu(g_j)} + \frac{\mu(fg_j)}{\nu(g_j)} - \frac{\mu(fg_j)}{\mu(g_j)} \quad (4.32b)$$

$$= \frac{\kappa^{-1}}{\nu(g_j)} [\nu(\kappa fg_j) - \mu(\kappa fg_j)] + \frac{\mu(fg_j)}{\mu(g_j)} \frac{\kappa^{-1}}{\nu(g_j)} [\mu(\kappa g_j) - \nu(\kappa g_j)]. \quad (4.32c)$$

Now notice that $\nu(g_j)^{-1} \leq \kappa^{-1}$ and that $\mu(fg_j)/\mu(g_j) \leq 1$, since the expression corresponds to an expectation with respect to measure found from μ by reweighting with likelihood proportional to g_j . Thus

$$|(L_j \nu)(f) - (L_j \mu)(f)| \leq \kappa^{-2} |\nu(\kappa fg_j) - \mu(\kappa fg_j)| + \kappa^{-2} |\mu(\kappa g_j) - \nu(\kappa g_j)|.$$

Since $|\kappa g_j|_\infty \leq 1$, it follows that $|\kappa f g_j|_\infty$ and hence that

$$\mathbb{E}^\omega |(L_j \nu)(f) - (L_j \mu)(f)|^2 \leq 4\kappa^{-4} \sup_{|h|_\infty \leq 1} \mathbb{E}^\omega |\nu(h) - \mu(h)|^2.$$

The desired result follows. \square

4.3.3. Improved Proposals

In the particle filter described in the previous section, we propose according to the underlying unobserved dynamics, and then apply Bayes's formula to incorporate the data. The final point in Remark 4.6 demonstrates that this may result in a very poor set of particles with which to approximate the filtering distribution. Cleverer proposals, which use the data, can lead to improved performance, and we outline this methodology here.

Instead of moving the particles $\{v_j^{(n)}\}_{n=1}^N$ according to the Markov kernel P , we use a Markov kernel Q_j with density $\mathbb{Q}(v_{j+1}|v_j, Y_{j+1})$. The weights $w_{j+1}^{(n)}$ are found, as before, by applying Bayes's formula for each particle and then weighting appropriately as in (4.27):

$$\tilde{w}_{j+1}^{(n)} = w_j^{(n)} \frac{\mathbb{P}\left(y_{j+1}|\hat{v}_{j+1}^{(n)}\right) \mathbb{P}\left(\hat{v}_{j+1}^{(n)}|v_j^{(n)}\right)}{\mathbb{Q}\left(\hat{v}_{j+1}^{(n)}|v_j^{(n)}, Y_{j+1}\right)}, \quad (4.33a)$$

$$w_{j+1}^{(n)} = \tilde{w}_{j+1}^{(n)} / \left(\sum_{n=1}^N \tilde{w}_{j+1}^{(n)} \right). \quad (4.33b)$$

The choice

$$\mathbb{Q}\left(v_{j+1}|v_j^{(n)}, Y_{j+1}\right) = \mathbb{P}\left(v_{j+1}|v_j^{(n)}\right)$$

results in the bootstrap filter from the preceding subsection. In the more general case, the approach results in the following algorithm:

1. Set $j = 0$ and $\mu_0^N(v_0)dv_0 = \mathbb{P}(v_0)dv_0$.
2. Draw $v_j^{(n)} \sim \mu_j^N$, $n = 1, \dots, N$.
3. Set $w_j^{(n)} = 1/N$, $n = 1, \dots, N$.
4. Draw $\hat{v}_{j+1}^{(n)} \sim \mathbb{Q}(\cdot|v_{j+1}^{(n)}, Y_{j+1})$.
5. Define $w_{j+1}^{(n)}$ by (4.33) and $\mu_{j+1}^N = \mathbb{P}^N(v_{j+1}|Y_{j+1})$ by (4.26).
6. $j + 1 \rightarrow j$.
7. Go to step 2.

We note that the normalization constants in (4.33a), here assumed known in the definition of the reweighting, are not, of course, needed. The so-called **optimal proposal** is found by choosing

$$\mathbb{Q}\left(v_{j+1}|v_j^{(n)}, Y_{j+1}\right) \equiv \mathbb{P}\left(v_{j+1}|v_j^{(n)}, y_{j+1}\right),$$

which results in

$$\tilde{w}_{j+1}^{(n)} = w_j^{(n)} \mathbb{P}\left(y_{j+1}|v_j^{(n)}\right). \quad (4.34)$$

The above can be seen by observing that the definition of conditional probability gives

$$\begin{aligned} \mathbb{P}\left(y_{j+1}|\hat{v}_{j+1}^{(n)}\right)\mathbb{P}\left(\hat{v}_{j+1}^{(n)}|v_j^{(n)}\right) &= \mathbb{P}\left(y_{j+1}, \hat{v}_{j+1}^{(n)}|v_j^{(n)}\right) \\ &= \mathbb{P}\left(\hat{v}_{j+1}^{(n)}|v_j^{(n)}, y_{j+1}\right)\mathbb{P}\left(y_{j+1}|v_j^{(n)}\right). \end{aligned} \quad (4.35)$$

Substituting the optimal proposal into (4.33) then immediately gives (4.34).

This small difference from the bootstrap filter may seem trivial at first glance, and at the potentially large cost of sampling from \mathbb{Q} . However, in the case of nonlinear Gaussian Markov models that we study here, the distribution and the weights are given in closed form. If the dynamics is highly nonlinear or the model noise is larger than the observational noise, then the variance of the weights for the optimal proposal may be much smaller than for the standard proposal. The corresponding particle filter will be referred to with the acronym SIRS(OP) to indicate the optimal proposal. For deterministic dynamics, the optimal proposal reduces to the standard proposal.

4.4 Large-Time Behavior of Filters

With the exception of the Kalman filter for linear problems and the particle filter in the general case, the filtering methods presented in this chapter do not, in general, give accurate approximations of the true posterior distribution; in particular, the approximate Gaussian filters do not perform well as measured by the Bayesian quality assessment test of Section 2.7. However, they may perform well as measured by the signal estimation quality assessment test, and the purpose of this section is to demonstrate this fact.

More generally, an important question concerning filters is their behavior when iterated over long times and in particular, their ability to recover the true signal underlying the data if iterated for long enough, even when initialized far from the truth. In this section, we present some basic large-time asymptotic results for filters to illustrate the key issue that affects the ability of filters to accurately recover the signal when iterated for long enough. The main idea is that the data must be sufficiently rich to stabilize any inherent instabilities within the underlying dynamical model (2.1); in rough terms, it is necessary to observe only the unstable directions, since the dynamics of the model itself will enable recovery of the true signal within the space spanned by the stable directions. We illustrate this idea first, in Section 4.4.1, for the explicitly solvable case of the Kalman filter in one dimension, and then, in Section 4.4.2, for the 3DVAR method.

4.4.1. The Kalman Filter in One Dimension

We consider the case of one-dimensional dynamics with

$$\Psi(v) = \lambda v, \quad h(v) = v,$$

while we will also assume that

$$\Sigma = \sigma^2, \quad \Gamma = \gamma^2.$$

With these definitions, equations (4.3a,b) become

$$\frac{1}{c_{j+1}} = \frac{1}{\sigma^2 + \lambda^2 c_j} + \frac{1}{\gamma^2}, \quad (4.36a)$$

$$\frac{m_{j+1}}{c_{j+1}} = \frac{\lambda m_j}{\sigma^2 + \lambda^2 c_j} + \frac{1}{\gamma^2} y_{j+1}, \quad (4.36b)$$

which, after some algebraic manipulation, give

$$c_{j+1} = g(c_j), \quad (4.37a)$$

$$m_{j+1} = \left(1 - \frac{c_{j+1}}{\gamma^2}\right) \lambda m_j + \frac{c_{j+1}}{\gamma^2} y_{j+1}, \quad (4.37b)$$

where we have defined

$$g(c) := \frac{\gamma^2(\lambda^2 c + \sigma^2)}{\gamma^2 + \lambda^2 c + \sigma^2}. \quad (4.38)$$

We wish to study the behavior of the Kalman filter as $j \rightarrow \infty$, i.e., when more and more data points are assimilated into the model. Note that the covariance evolves independently of the data $\{y_j\}_{j \in \mathbb{Z}^+}$ and satisfies an autonomous nonlinear dynamical system. However, it is of interest to note that if $\sigma^2 = 0$, then the dynamical system for c_j^{-1} is linear.

We now study the asymptotic properties of this map. The fixed points c^* of (4.37a) satisfy

$$c^* = \frac{\gamma^2(\lambda^2 c^* + \sigma^2)}{\gamma^2 + \lambda^2 c^* + \sigma^2}, \quad (4.39)$$

and thus solve the quadratic equation

$$\lambda^2(c^*)^2 + (\gamma^2(1 - \lambda^2) + \sigma^2)c^* - \gamma^2\sigma^2 = 0.$$

We see that provided $\lambda\gamma\sigma \neq 0$, one root is positive and one negative. The roots are given by

$$c_{\pm}^* = \frac{-(\gamma^2 + \sigma^2 - \gamma^2\lambda^2) \pm \sqrt{(\gamma^2 + \sigma^2 - \gamma^2\lambda^2)^2 + 4\lambda^2\gamma^2\sigma^2}}{2\lambda^2}. \quad (4.40)$$

We observe that the update formula for the covariance ensures that provided $c_0 \geq 0$, then $c_j \geq 0$ for all $j \in \mathbb{N}$. It also demonstrates that $c_j \leq \gamma^2$ for all $j \in \mathbb{Z}^+$, so that the variance of the filter is no larger than the variance in the data. We may hence fix our attention on nonnegative covariances, knowing that they are also uniformly bounded by γ^2 . We will now study the stability of the nonnegative fixed points.

We begin with the case $\sigma = 0$, which corresponds to deterministic dynamics and for which the dynamics of c_j^{-1} is linear. In this case, we obtain

$$c_+^* = 0, \quad c_-^* = \frac{\gamma^2(\lambda^2 - 1)}{\lambda^2}$$

and

$$g'(c_+^*) = \lambda^2, \quad g'(c_-^*) = \lambda^{-2},$$

which implies that when $\lambda^2 < 1$, c_+^* is an asymptotically stable fixed point, while when $\lambda^2 > 1$, c_-^* is an asymptotically stable fixed point. When $|\lambda| = 1$, the two roots are coincident at the origin and neutrally stable. Using the aforementioned linearity, for the case $\sigma = 0$ it is possible to solve (4.36a) to obtain for $\lambda^2 \neq 1$,

$$\frac{1}{c_j} = \left(\frac{1}{\lambda^2} \right)^j \frac{1}{c_0} + \frac{1}{\gamma^2} \left[\frac{\left(\frac{1}{\lambda^2} \right)^j - 1}{\frac{1}{\lambda^2} - 1} \right]. \quad (4.41)$$

This explicit formula shows that the fixed point c_+^* (respectively c_-^*) is globally asymptotically stable, and exponentially attracting on \mathbb{R}^+ , when $\lambda^2 < 1$ (respectively $\lambda^2 > 1$). Notice also that $c_-^* = \mathcal{O}(\gamma^2)$, so that when $\lambda^2 > 1$, the asymptotic variance of the filter scales as the observational noise variance. Furthermore, when $\lambda^2 = 1$, we may solve (4.36a) to obtain

$$\frac{1}{c_j} = \frac{1}{c_0} + \frac{j}{\gamma^2},$$

showing that $c_-^* = c_+^* = 0$ is globally asymptotically stable on \mathbb{R}^+ but is only algebraically attracting.

We now study the stability of the fixed points c_+^* and c_-^* in the case of $\sigma^2 > 0$, corresponding to the case in which the dynamics are stochastic. To this end, we prove some bounds on $g'(c^*)$ that will also be useful when we study the behavior of the error between the true signal and the estimated mean; here, and in what follows in the remainder of this example, a prime denotes differentiation with respect to c . We begin by noting that

$$g(c) = \gamma^2 - \frac{\gamma^4}{\gamma^2 + \lambda^2 c + \sigma^2}, \quad (4.42)$$

and so

$$g'(c) = \frac{\lambda^2 \gamma^4}{(\gamma^2 + \lambda^2 c + \sigma^2)^2}.$$

Using the fact that c^* satisfies (4.39) together with equation (4.42), we obtain

$$g'(c^*) = \frac{1}{\lambda^2} \frac{(c^*)^2}{\left(c^* + \frac{\sigma^2}{\lambda^2} \right)^2} \quad \text{and} \quad g'(c^*) = \lambda^2 \left(1 - \frac{c^*}{\gamma^2} \right)^2.$$

We can now see that from the first equation, we obtain the following two bounds, since $\sigma^2 > 0$:

$$g'(c^*) < \lambda^{-2}, \quad \text{for } \lambda \in \mathbb{R}, \quad \text{and} \quad g'(c^*) < 1, \quad \text{for } \lambda^2 = 1,$$

while from the second equality and the fact that since c^* satisfies (4.39), $c^* < \gamma^2$, we obtain

$$g'(c^*) < \lambda^2$$

when $c^* > 0$. We thus conclude that when $\sigma^2 > 0$, the fixed point c_+^* of (4.37a) is always stable independently of the value of the parameter λ .

	Limiting covariance for $\sigma^2 = 0$	Limiting covariance for $\sigma^2 > 0$
$ \lambda < 1$	$c_j \rightarrow 0$ (exponentially)	$c_j \rightarrow c_+^* = \mathcal{O}(\gamma^2)$ (exponentially)
$ \lambda = 1$	$c_j \rightarrow 0$ (algebraically)	$c_j \rightarrow c_+^* = \mathcal{O}(\gamma^2)$ (exponentially)
$ \lambda > 1$	$c_j \rightarrow c_-^* = \mathcal{O}(\gamma^2)$ (exponentially)	$c_j \rightarrow c_+^* = \mathcal{O}(\gamma^2)$ (exponentially)

Table 4.1: Summary of the limiting behavior of covariance c_j for Kalman filter applied to one-dimensional dynamics.

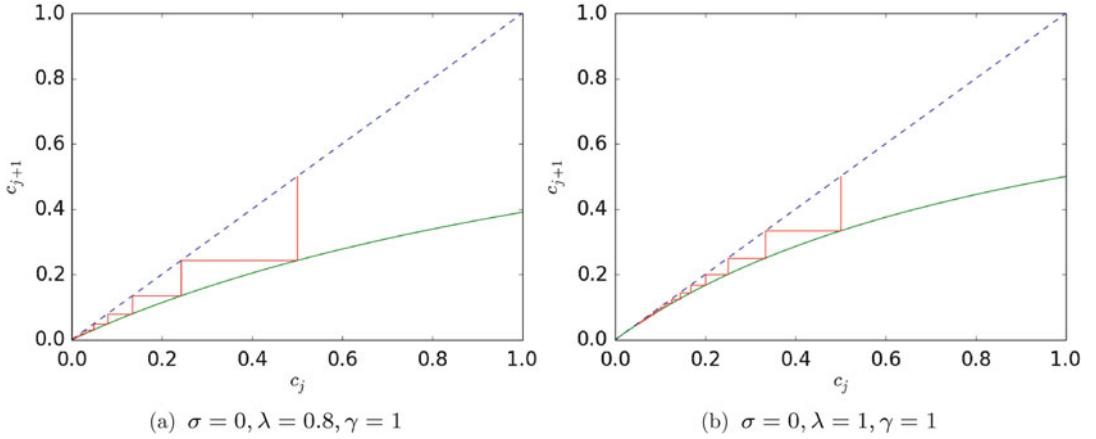


Fig. 4.1: Cobweb diagram for equation (4.37a).

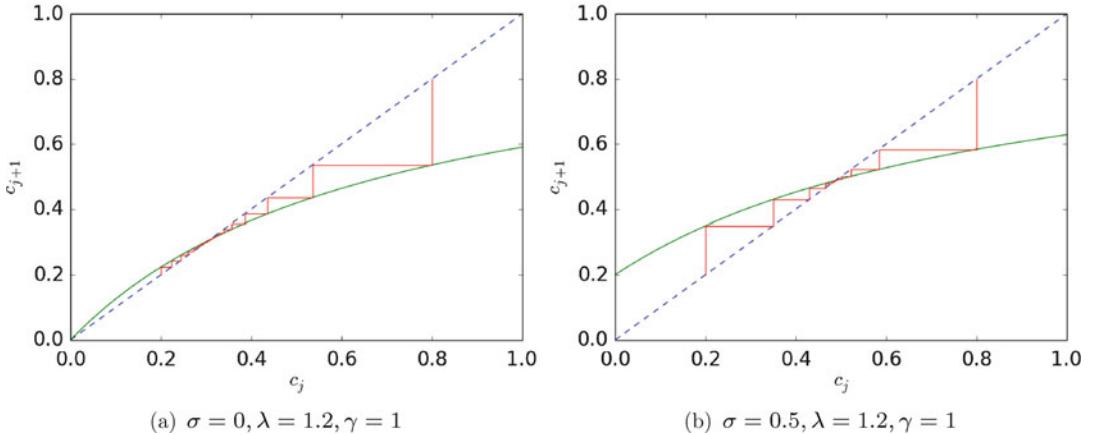


Fig. 4.2: Cobweb diagram for equation (4.37a).

Table 4.1 summarizes the behavior of the variance of the Kalman filter in the case of one-dimensional dynamics. This is illustrated further in Figures 4.1 and 4.2, where we plot the cobweb diagram for the map (4.42). In particular, in Figure 4.1, we observe the difference between the algebraic and geometric convergence to 0, for different values of λ in the case $\sigma = 0$, while in Figure 4.2, we observe the exponential convergence to c_+^* for the case $|\lambda| > 1$. The analysis of the error between the mean and the truth underlying the data is left as an exercise at the end of the chapter. This shows that the error in the mean is, asymptotically, of order γ^2 in the case $\sigma = 0$.

4.4.2. The 3DVAR Filter

In the previous subsection, we showed that the Kalman filter accurately recovers a one-dimensional signal, provided the observational noise is small. The result allows for initialization far from the true signal and is, in this sense, quite strong. On the other hand, being only one-dimensional, it gives a somewhat limited picture. In this subsection, we study the 3DVAR filter given by (4.14). We study conditions under which the 3DVAR filter will recover the true signal, to within a small observational noise level of accuracy, in dimensions greater than 1, and when only part of the system is observed.

To this end, we assume that

$$y_{j+1} = Hv_{j+1}^\dagger + \epsilon_j, \quad (4.43)$$

where the true signal $\{v_j^\dagger\}_{j \in \mathbb{N}}$ satisfies

$$v_{j+1}^\dagger = \Psi(v_j^\dagger), \quad j \in \mathbb{N}, \quad (4.44a)$$

$$v_0^\dagger = u, \quad (4.44b)$$

and for simplicity, we assume that the observational noise satisfies

$$\sup_{j \in \mathbb{N}} |\epsilon_j| = \epsilon. \quad (4.45)$$

We have the following result.

Theorem 4.10. *Assume that the data is given by (4.43), where the signal follows equation (4.44) and the error in the data satisfies (4.45). Assume furthermore that \widehat{C} is chosen such that $(I - KH)\Psi : \mathbb{R}^n \rightarrow \mathbb{R}^n$ is globally Lipschitz with constant $a < 1$ in some norm $\|\cdot\|$. Then there is constant $c > 0$ such that*

$$\limsup_{j \rightarrow \infty} \|m_j - v_j^\dagger\| \leq \frac{c}{1-a}\epsilon.$$

Proof We may write (4.14), (4.44), using (4.43), as

$$\begin{aligned} m_{j+1} &= (I - KH)\Psi(m_j) + KH\Psi(v_j^\dagger) + K\epsilon_j, \\ v_{j+1}^\dagger &= (I - KH)\Psi(v_j^\dagger) + KH\Psi(v_j^\dagger). \end{aligned}$$

Subtracting, and letting $e_j = m_j - v_j^\dagger$, gives, for some finite constant c independent of j ,

$$\begin{aligned} \|e_{j+1}\| &\leq \|(I - KH)\Psi(m_j) - (I - KH)\Psi(v_j^\dagger)\| + \|K\epsilon_j\| \\ &\leq a\|e_j\| + c\epsilon. \end{aligned}$$

Applying the Gronwall lemma (Lemma 1.14) gives the desired result. \square

We refer to a map with Lipschitz constant less than 1 as a *contraction* in what follows.

Remark 4.11. *The preceding simple theorem shows that it is possible to construct filters that can recover from being initialized far from the truth and lock on to a small neighborhood of the true signal underlying the data when run for long enough. Furthermore, this can happen even when the system is only partially observed, provided that the observational noise is small and enough of the system is observed. This concept of observing “enough” illustrates a key idea in filtering: the question whether the fixed model covariance in 3DVAR, \widehat{C} , can be chosen to make*

$(I - KH)\Psi$ into a contraction involves a subtle interplay between the underlying dynamics, encapsulated in Ψ , and the observation operator H . In rough terms, the question of making $(I - KH)\Psi$ into a contraction is the question whether the unstable parts of the dynamics are observed; if they are, then it is typically the case that \widehat{C} can be designed to obtain the desired contraction.



Example 4.12. Assume that $H = I$, so that the whole system is observed, that $\Gamma = \gamma^2 I$ and $\widehat{C} = \sigma^2 I$. Then for $\eta^2 = \frac{\gamma^2}{\sigma^2}$,

$$S = (\sigma^2 + \gamma^2)I, \quad K = \frac{\sigma^2}{(\sigma^2 + \gamma^2)}I$$

and

$$(I - KH) = \frac{\gamma^2}{(\sigma^2 + \gamma^2)}I = \frac{\eta^2}{(1 + \eta^2)}I.$$

Thus if $\Psi : \mathbb{R}^n \rightarrow \mathbb{R}^n$ is globally Lipschitz with constant $\lambda > 0$ in the Euclidean norm, $|\cdot|$, then $(I - KH)\Psi$ is globally Lipschitz with constant $a < 1$ if η is chosen such that $\frac{\eta^2 \lambda}{1 + \eta^2} < 1$. Thus by choosing η sufficiently small, the filter can be made to contract. This corresponds to trusting the data sufficiently in comparison to the model. It is a form of **variance inflation** in that for a given level of observational noise, η can be made sufficiently small by choosing the model variance scale σ^2 sufficiently large—“inflating” the model variance.



Example 4.13. Assume that there is a partition of the state space in which $H = (I, 0)^T$, so that only part of the system is observed. Set $\Gamma = \gamma^2 I$ and $\widehat{C} = \sigma^2 I$. Then with η as in the previous example,

$$I - KH = \begin{pmatrix} \frac{\eta^2}{1 + \eta^2} I & 0 \\ 0 & I \end{pmatrix}.$$

While the previous example shows that variance inflation may help to stabilize the filter, this example shows that in general, more is required: in this case, it is clear that making $(I - KH)\Psi(\cdot)$ into a contraction will require a relationship between the subspace in which we observe and the space in which the dynamics of the map is expanding and contracting. For example, if $\Psi(u) = Lu$ and

$$L = \begin{pmatrix} 2I & 0 \\ 0 & aI \end{pmatrix},$$

then

$$(I - KH)L = \begin{pmatrix} \frac{2\eta^2}{1 + \eta^2} I & 0 \\ 0 & aI \end{pmatrix}.$$

When $|a| < 1$, this can be made into a contraction by choosing η sufficiently small; but for $|a| \geq 1$, this is no longer possible. The example thus illustrates the intuitive idea that the observations should be sufficiently rich to ensure that the unstable directions within the dynamics can be tamed by observing them.



4.4.3. The Synchronization Filter

A fundamental idea underlying successful filtering of partially observed dynamical systems is synchronization. To illustrate this, we introduce and study the idealized *synchronization filter*. To this end, consider a partition of the identity $P + Q = I$. We write $v = (p, q)$, where $p = Pv$, $q = Qv$, and then, with a slight abuse of notation, write $\Psi(v) = \Psi(p, q)$. Consider a true signal governed by the deterministic dynamics model (4.44) and write $v_k^\dagger = (p_k^\dagger, q_k^\dagger)$, with $p_k^\dagger = Pv_k^\dagger$ and $q_k^\dagger = Qv_k^\dagger$. Then

$$\begin{aligned} p_{k+1}^\dagger &= P\Psi(p_k^\dagger, q_k^\dagger), \\ q_{k+1}^\dagger &= Q\Psi(p_k^\dagger, q_k^\dagger). \end{aligned}$$

Now imagine that we observe $y_k = p_k^\dagger$ exactly, without noise. Then the synchronization filter simply fixes the image under P to p_k^\dagger and plugs this into the image of the dynamical model under Q ; if the filter is $m_k = (p_k, q_k)$ with $p_k = Pm_k$ and $q_k = Qm_k$, then

$$\begin{aligned} p_{k+1} &= p_{k+1}^\dagger, \\ q_{k+1} &= Q\Psi(p_k^\dagger, q_k). \end{aligned}$$

We note that expressed in terms of the data, this filter has the form

$$m_{k+1} = Q\Psi(m_k) + Py_{k+1}. \quad (4.46)$$

A key question now is whether the filter synchronizes in the following sense:

$$|q_k - q_k^\dagger| \rightarrow 0 \text{ as } k \rightarrow \infty.$$

This of course is equivalent to

$$|m_k - v_k^\dagger| \rightarrow 0 \text{ as } k \rightarrow \infty. \quad (4.47)$$

Whether this happens involves, as for 3DVAR described above, a subtle interplay between the underlying dynamics and the observation operator, here P . The bibliography, Section 4.6, contains pointers to the literature studying this question.

In fact, the following example shows how the synchronization filter can be viewed as a distinguished parameter limit, corresponding to infinite variance inflation, for a particular family of 3DVAR filters.

Example 4.14. Let $H = P$ and $\Gamma = \gamma^2 I$. If we choose \hat{C} as in Example 4.13, then the 3DVAR filter can be written as

$$m_{k+1} = S\Psi(m_k) + (I - S)y_{k+1}, \quad (4.48a)$$

$$S = \frac{\eta^2}{1 + \eta^2} P + Q. \quad (4.48b)$$

The limit $\eta \rightarrow 0$ is the extreme limit of variance inflation referred to in Example 4.12. In this limit, the 3DVAR filter becomes the synchronization filter (4.46). ♠

4.5 Illustrations

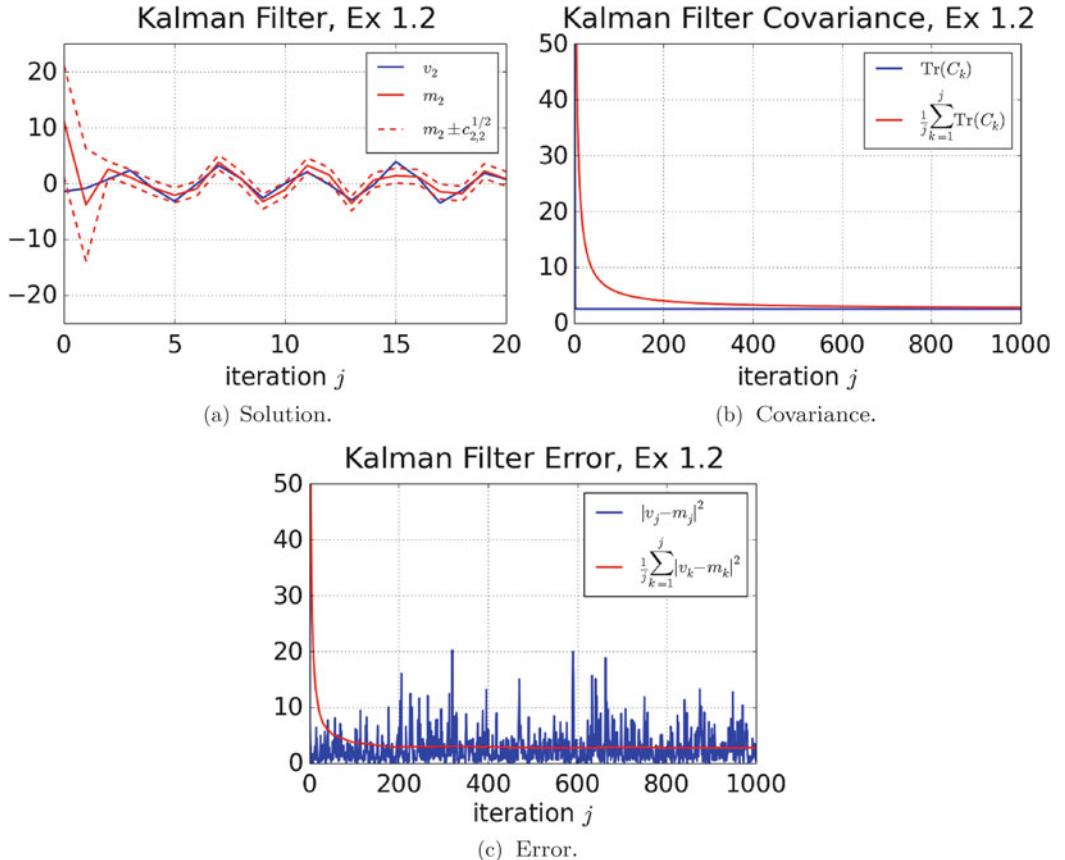


Fig. 4.3: Kalman filter applied to the linear system of Example 2.2 with $A = A_3$, $H = (1, 0)$, $\Sigma = I$, and $\Gamma = 1$; see also p8.m in Section 5.2.5. The problem is initialized with mean 0 and covariance $10I$.

The first illustration concerns the Kalman filter applied to the linear system of Example 2.3 with $A = A_3$. We assume that $H = (1, 0)$, so that we observe only the first component of the system, and the model and observational covariances are $\Sigma = I$ and $\Gamma = 1$, where I is the 2×2 identity matrix. The problem is initialized with mean 0 and covariance $10I$. Figure 4.3a shows the behavior of the filter on the unobserved component, showing how the mean locks onto a small neighborhood of the truth and how the one-standard-deviation confidence intervals computed from the variance on the second component also shrink from a large initial value to an asymptotic small value; this value is determined by the observational noise variance in the first component. In Figure 4.3b, the trace of the covariance matrix is plotted, demonstrating that the total covariance matrix approaches a small limiting matrix asymptotically. And finally, Figure 4.3c shows the error (in the Euclidean norm) between the filter mean and the truth underlying the data, together with its running average. We will employ similar figures (a), (b), and (c) in the examples that follow in this section.

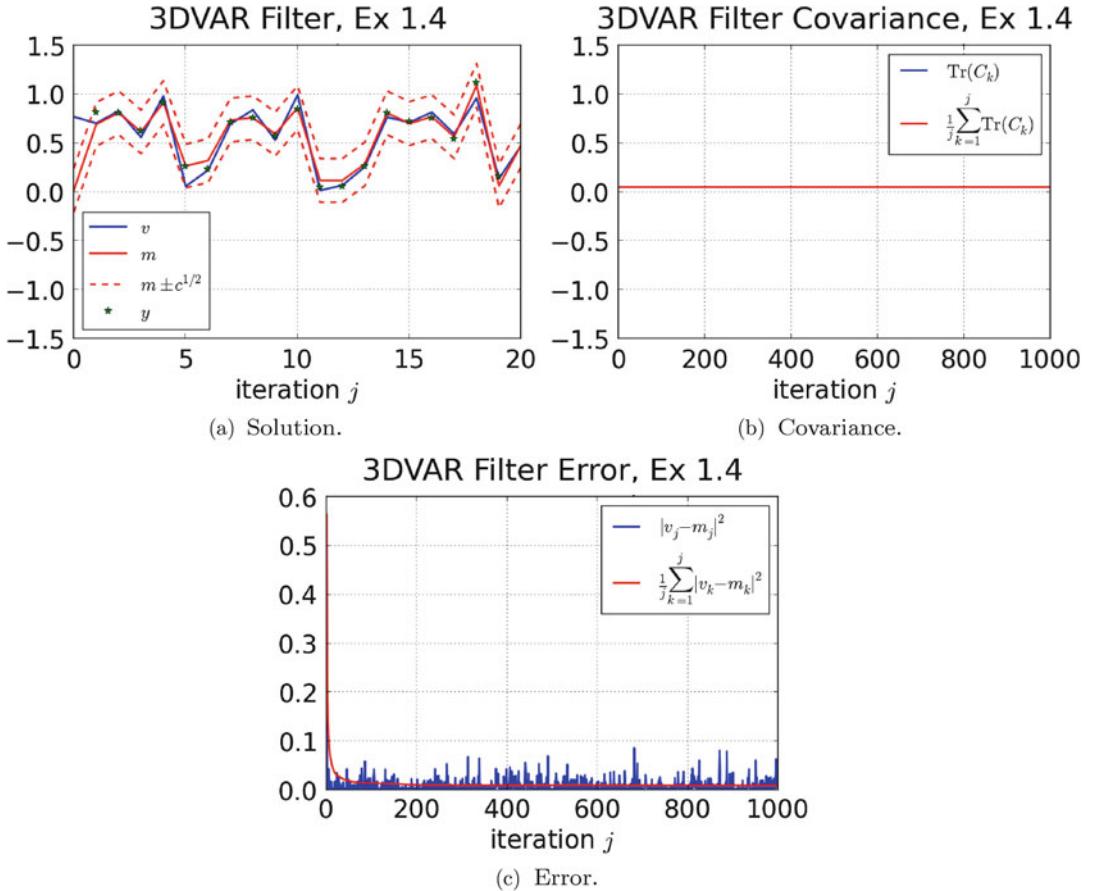


Fig. 4.4: 3DVAR methodology applied to the logistic map Example 2.4 with $r = 4$, $\gamma^2 = 10^{-2}$, and $c = \gamma^2/\eta$ with $\eta = 0.2$; see also p9.m in Section 5.3.1.

The next illustration shows the 3DVAR algorithm applied to Example 2.4 with $r = 2.5$. We consider noise-free dynamics and observational variance of $\gamma^2 = 10^{-2}$. The fixed model covariance is chosen to be $c = \gamma^2/\eta$ with $\eta = 0.2$. The resulting algorithm performs well at tracking the truth with asymptotic time-averaged Euclidean error of size roughly 10^{-2} . See Figure 4.4.

The rest of the figures illustrate the behavior of the various filters, all applied to Example 2.3 with $\alpha = 2.5$, $\sigma = 0.3$, and $\gamma = 1$. In particular, 3DVAR (Figure 4.5), ExKF (Figure 4.6), EnKF (Figure 4.7), ETKF (Figure 4.8), and the particle filter with standard (Figure 4.9) and optimal (Figure 4.10) proposals are all compared on the same example. The ensemble-based methods all use 100 ensemble members each (notice that this is much larger than the dimension of the state space, which is $n = 1$ here; this is hence a regime outside of which the ensemble methods would usually be employed in practice). For 3DVAR, results from which (for this example) are shown only in the summary Figure 4.11, we take $\eta = 0.5$.

All of the methods perform well at tracking the true signal, asymptotically in time, recovering from a large initial error. However, they also all exhibit occasional instabilities and lose track of the true signal for short periods of time. From Fig. 4.6(c), we can observe that

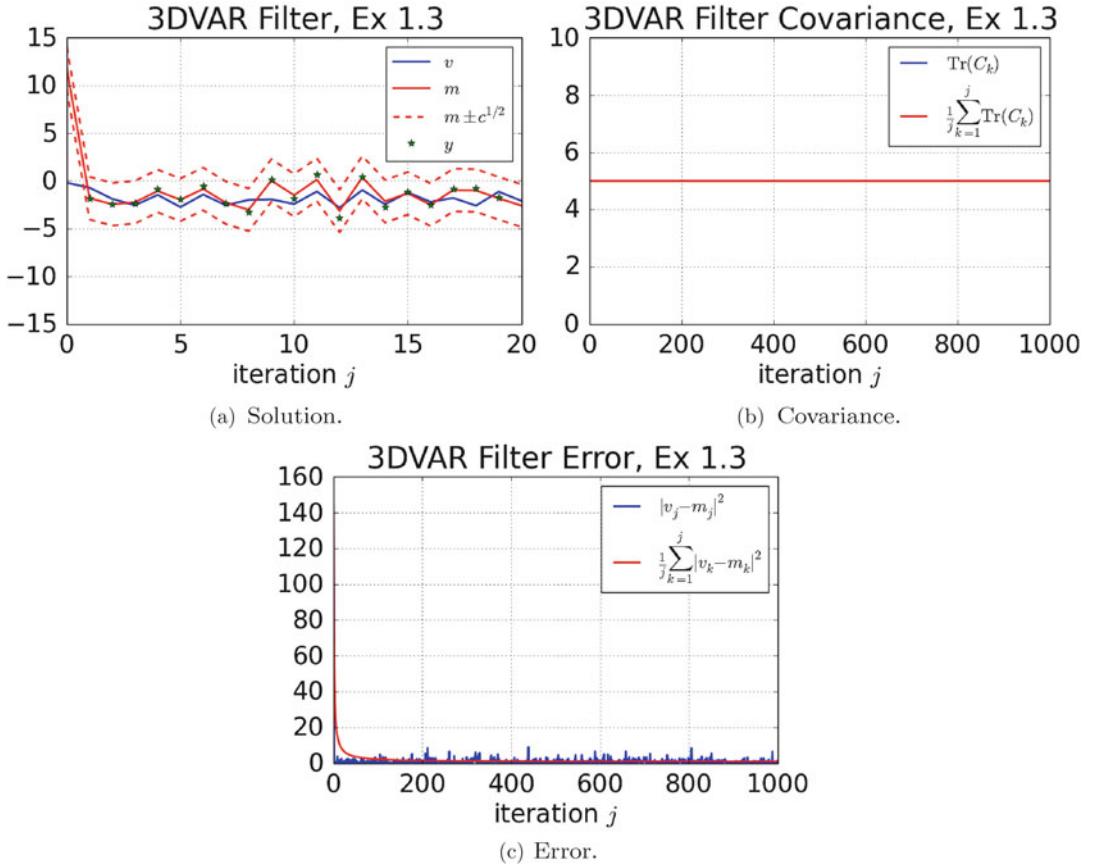


Fig. 4.5: 3DVAR for the sine map Example 2.3 with $\alpha = 2.5$, $\sigma = 0.3$, $\gamma = 1$, and $\eta = 0.2$; see also p10.m in Section 5.3.3.

the ExKF has small error for most of the simulation, but that sporadic large excursions are seen in the error. From Fig. 4.8(c), one can observe that ETKF is similarly prone to small destabilization and local instability, like the EnKF with perturbed observations in Fig. 4.7(c). Also, notice from Figure 4.9(c) that the particle filter with standard proposal is perhaps slightly more prone to destabilization than the optimal proposal in Figure 4.10(c), although the difference is minimal.

The performance of the filters is now compared through a detailed study of the statistical properties of the error $e = m - v^\dagger$ over long simulation times. In particular, we compare the histograms of the errors and their large time averages. Figure 4.11 compares the errors incurred by the three basic methods 3DVAR, ExKF, and EnKF, demonstrating that the EnKF is the most accurate method of the three on average, with ExKF the least accurate on average. Notice from Fig. 4.11(a) that the error distribution of 3DVAR is the widest, and both it and EnKF remain consistently accurate. The distribution of ExKF is similar to that of EnKF, except for the “fat tails” associated with the destabilization intervals seen in Fig. 4.6.

Figure 4.12 compares the errors incurred by the four more accurate ensemble-based methods EnKF, ETKF, SIRS, and SIRS(OP). The error distribution, Fig. 4.12(a), of all these filters is similar. In Fig. 4.12(b), one can see that the time-averaged errors in EnKF and ETKF are indistinguishable. Also, the EnKF, ETKF, and SIRS(OP) also remain more or less

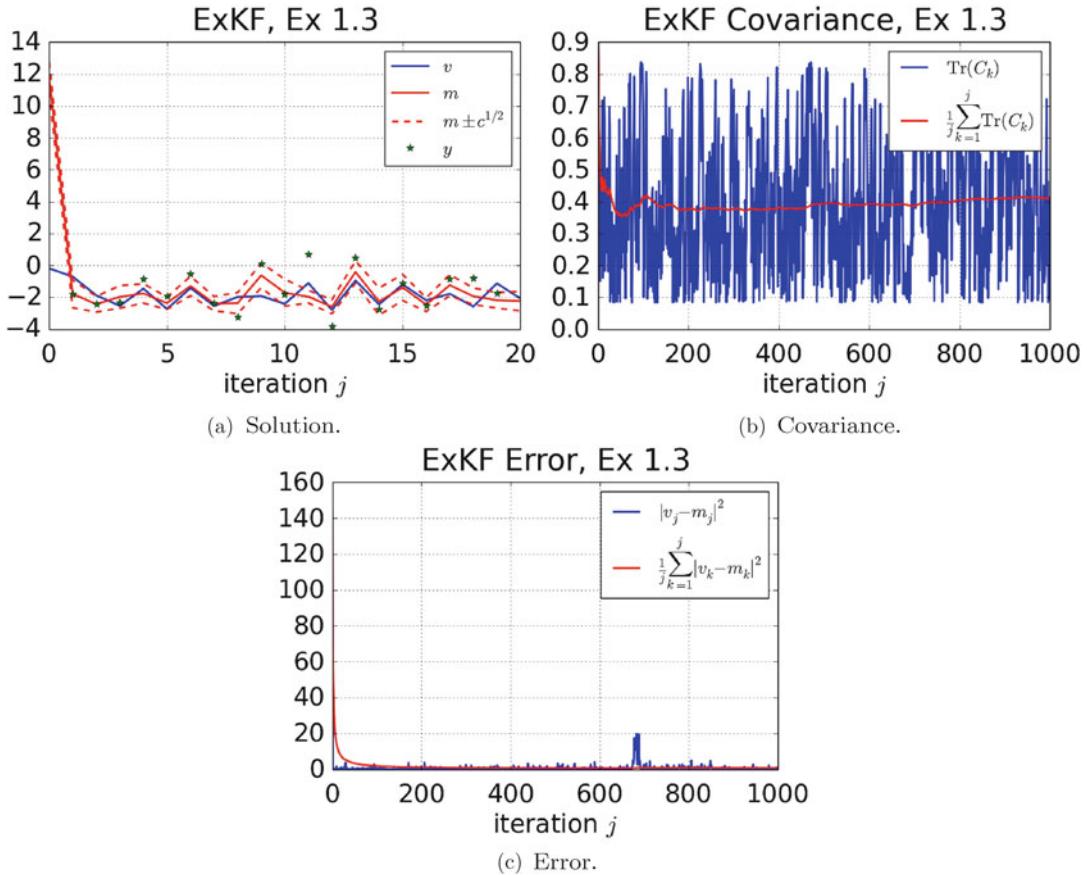


Fig. 4.6: ExKF on the sine map Example 2.3 with $\alpha = 2.5$, $\sigma = 0.3$, and $\gamma = 1$; see also p11.m in Section 5.3.4.

consistently accurate. The distribution of e for SIRS is similar to SIRS(OP), except for the fat tails associated with the destabilization intervals seen in Fig. 4.9, which leads to the larger time-averaged error seen in Fig. 4.12(b). In this sense, the distribution of e is similar to that for ExKF.

4.6 Bibliographic Notes

- Section 4.1. Since its introduction in 1960 [79], the Kalman filter has found wide-ranging application to low-dimensional engineering applications in which the linear Gaussian model is appropriate. In addition to the original motivation in control of flight vehicles, it has grown in importance in the fields of econometric time-series analysis and signal processing [65]. It is also important because it plays a key role in the development of the approximate Gaussian filters, which are the subject of Section 4.2. The idea behind the Kalman filter, to combine model and data optimally, is arguably one of the most important ideas in applied

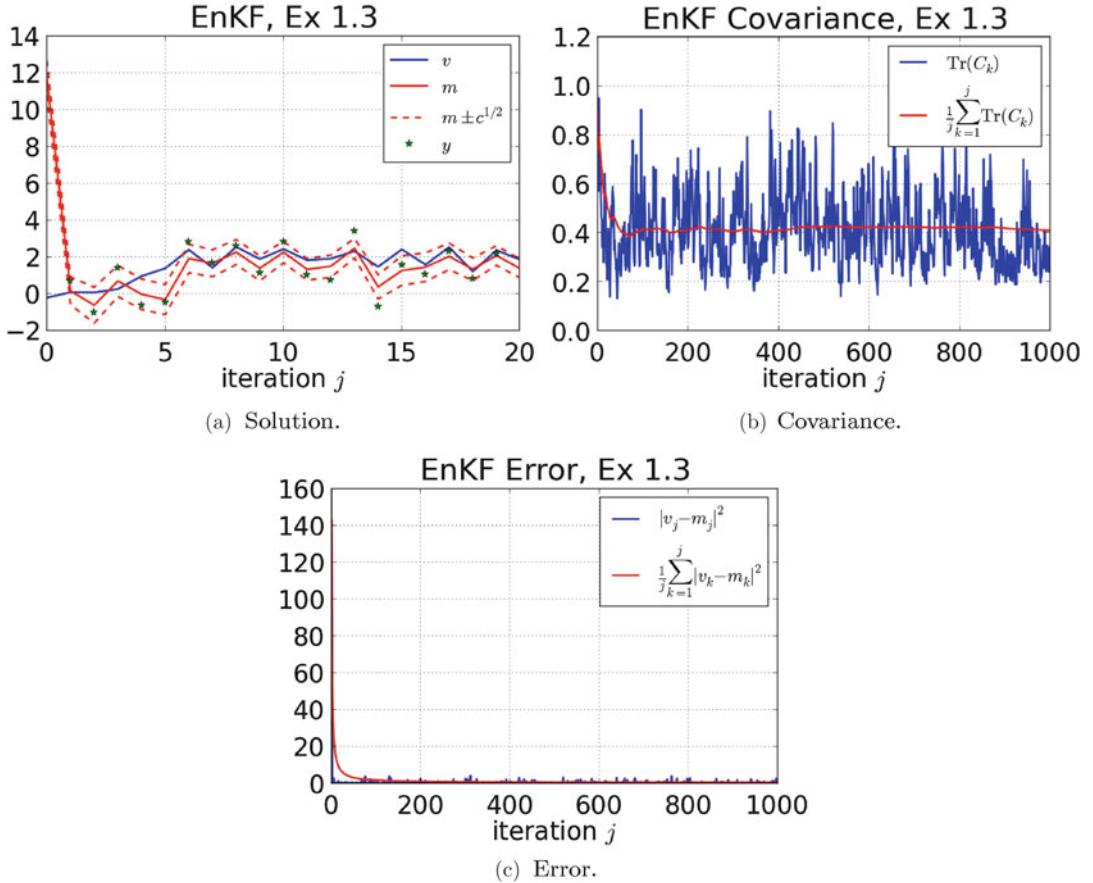


Fig. 4.7: EnKF on the sine map Example 2.3 with $\alpha = 2.5$, $\sigma = 0.3$, $\gamma = 1$, and $N = 100$; see also p12.m in Section 5.3.5.

mathematics over the last century: the impact of the paper [79] on many application domains has been huge.

- Section 4.2. All the non-Gaussian filters we discuss are based on modifying the Kalman filter so that it may be applied to nonlinear problems. The development of new filters is a very active area of research, and the reader is directed to the book [100], together with the articles [28, 101] and [142], for insight into some of the recent developments with an applied mathematics perspective.

The 3DVAR algorithm was proposed at the United Kingdom's Met Office in 1986 [93, 94], and was subsequently developed by the United States National Oceanic and Atmospheric Administration [119] and by the European Centre for Medium-Range Weather Forecasts (ECMWF) in [36]. The perspective of these papers was one of minimization, and as such, easily incorporates nonlinear observation operators via the objective functional (4.11), with a fixed $\hat{C} = \hat{C}_{j+1}$, for the analysis step of filtering; nonlinear observation operators are important in numerous applications, including numerical weather forecasting. In the case of linear observation operators, the objective functional is given by (4.12) with explicit solution given, in the case $\hat{C} = \hat{C}_{j+1}$, by (4.14). In fact, the method of *optimal interpolation* predates 3DVAR and takes the linear equations (4.14) as their starting point, rather than

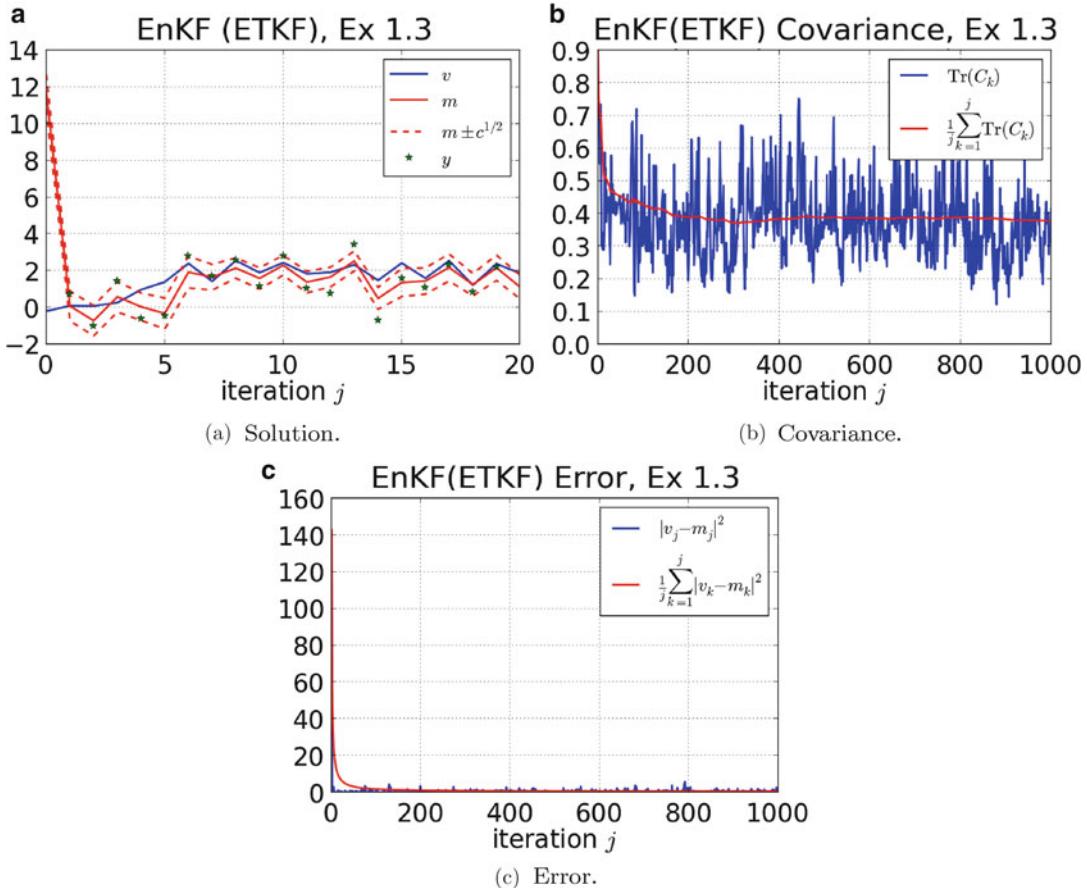


Fig. 4.8: ETKF on the sine map Example 2.3 with $\alpha = 2.5$, $\sigma = 0.3$, $\gamma = 1$, and $N = 100$; see also p13.m in Section 5.3.6.

starting from a minimization principle; it is then very closely related to the method of kriging from the geosciences [136]. The 3DVAR algorithm is important because it is prototypical of the many more sophisticated filters that are now widely used in practice, and it is thus natural to study it.

The extended Kalman filter was developed in the control theory community and is discussed at length in [77]. It is not practical to implement in high dimensions, and low-rank extended Kalman filters are then used instead; see [89] for a recent discussion.

The ensemble Kalman filter uses a set of particles to estimate covariance information, and may be viewed as an approximation of the extended Kalman filter, designed to be suitable in high dimensions. See [50] for an overview of the methodology, written by one of its originators, and [144] for an early example of the power of the method. We note that the minimization principle (4.15) has the very desirable property that the samples $\{\hat{v}_{n+1}^{(n)}\}_{n=1}^N$ correspond to samples of the Gaussian distribution found by Bayes's theorem with prior $N(\hat{m}_{j+1}, \hat{C}_{j+1})$ and likelihood $y_{j+1}|v$. This is the idea behind the *randomized maximum likelihood* method described in [115], widely used in petroleum applications; the idea is discussed in detail in the context of the EnKF in [82]. There has been some analysis of the EnKF in the large-sample limit; see, for example, [91, 90, 103]. However, the primary

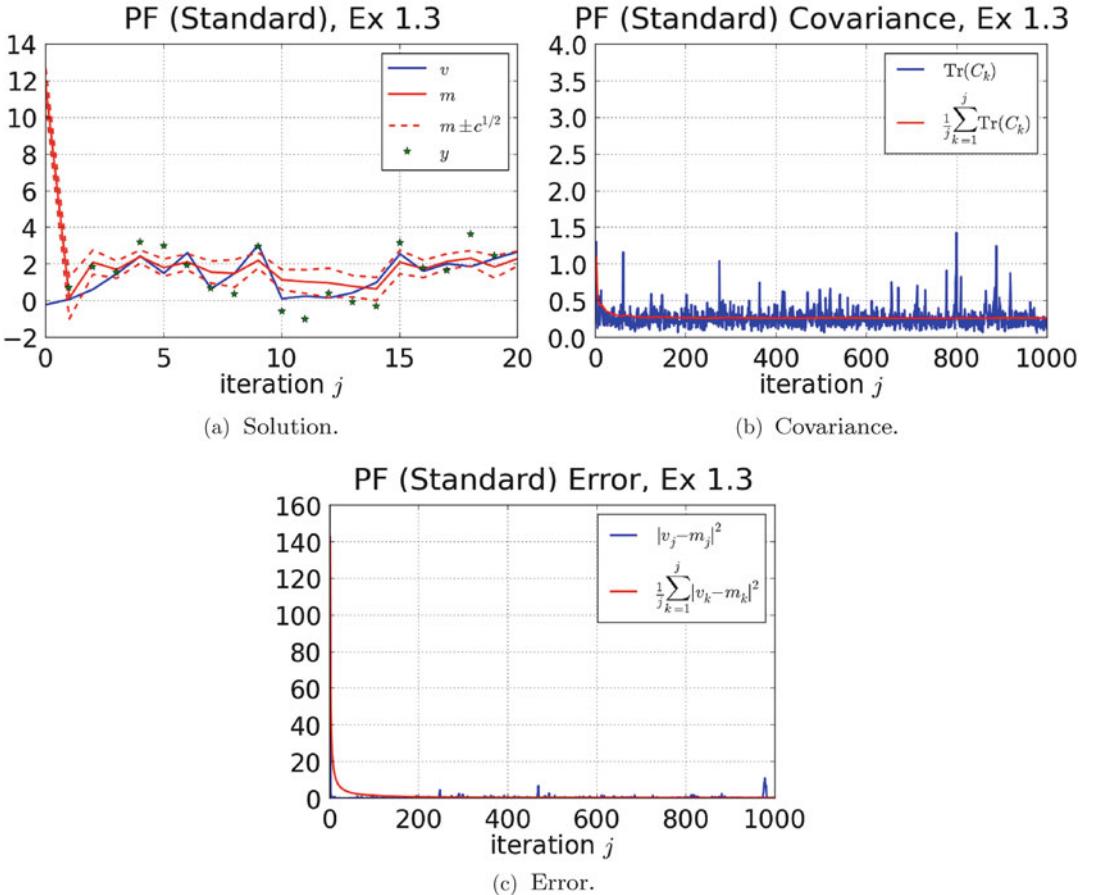


Fig. 4.9: Particle Filter (standard proposal) on the sine map Example 2.3 with $\alpha = 2.5$, $\sigma = 0.3$, $\gamma = 1$, and $N = 100$; see also p14.m in Section 5.3.7.

power of the method for practitioners is that it seems to provide useful information for small sample sizes; it is therefore perhaps a more interesting direction for analysis to study the behavior of the algorithm, and determine methodologies to improve it, for fixed numbers of ensemble members. There is some initial work in this direction, and we describe it below. Note that the Γ appearing in the perturbed observation EnKF can be replaced by the sample covariance $\tilde{\Gamma}$ of the $\{\eta_{j+1}^{(n)}\}_{n=1}^N$, and this is often done in practice. The sample covariance of the updated ensemble in this case is equal to $(I - \tilde{K}_{j+1}H)\hat{C}_{j+1}$, where \tilde{K}_{j+1} is the gain corresponding to the sample covariance $\tilde{\Gamma}$.

There is a range of parameters that can be used to tune the approximate Gaussian filters or modifications of those filters. In practical implementations, especially for high-dimensional problems, the basic forms of the ExKF and EnKF as described here are prone to poor behavior, and such tuning is essential [81, 50]. In Examples 4.12 and 4.13, we have already shown the role of variance inflation for 3DVAR, and this type of approach is also fruitfully used within ExKF and EnKF. A basic version of variance inflation is to replace the estimate \hat{C}_{j+1} in (4.13) by $\epsilon\hat{C} + \hat{C}_{j+1}$, where \hat{C} is a fixed covariance such as that used in a 3DVAR method. Introducing $\epsilon \in (0, 1)$ leads, for positive definite \hat{C} , to an operator without a null

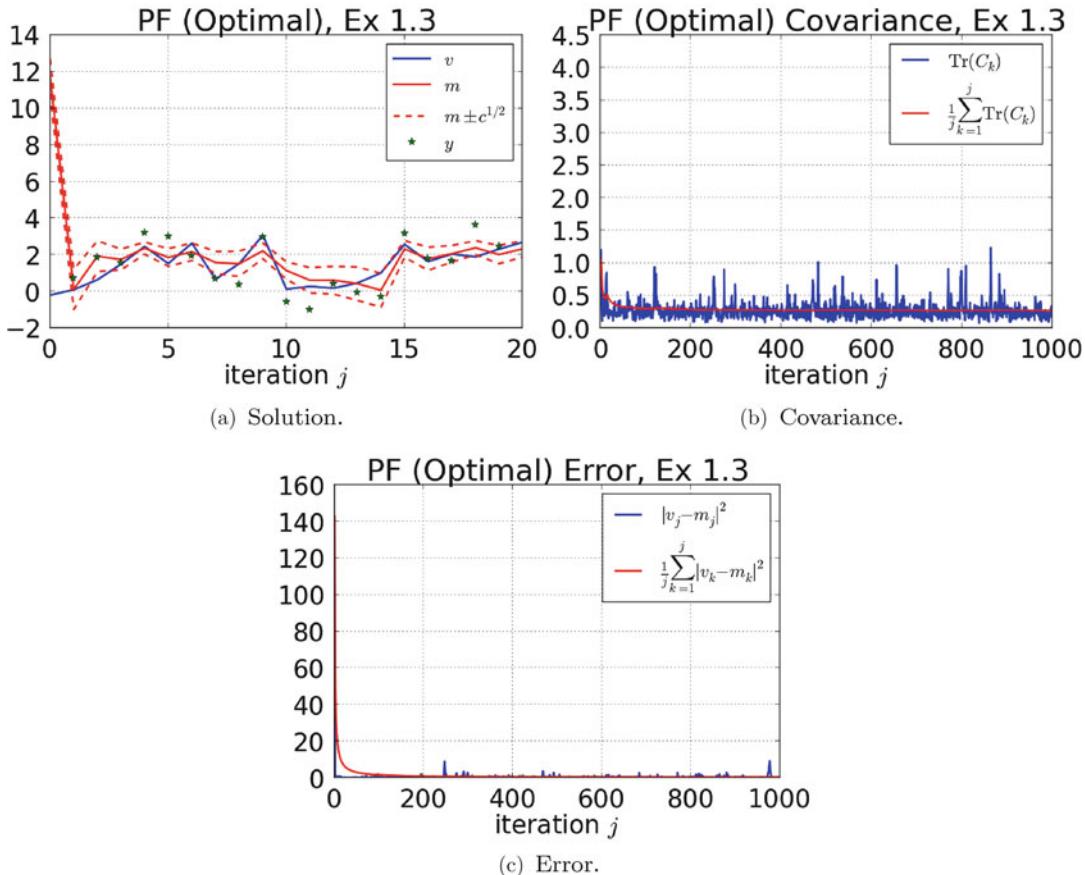


Fig. 4.10: Particle Filter (optimal proposal) on the sine map Example 1.3 with $\alpha = 2.5$, $\sigma = 0.3$, $\gamma = 1$, and $N = 100$; see also p15.m in Section 5.3.8.

space and consequently to better behavior. In contrast, taking $\epsilon = 0$ can lead to singular model covariances. This observation is particularly important when the EnKF is used in high-dimensional systems in which the number N of ensemble members is always less than the dimension n of the state space. In this situation, \hat{C}_{j+1} necessarily has a null space of dimension at least $n - N$. It can also be important for the ExKF, where the evolving dynamics can lead, asymptotically in j , to degenerate \hat{C}_{j+1} with nontrivial null space. Notice also that this form of variance inflation can be thought of as using 3DVAR-like covariance updates, in the directions not described by the ensemble covariance. This can be beneficial in terms of the ideas underlying Theorem 4.10, where the key idea is that K close to the identity can help ameliorate growth in the underlying dynamics. This may also be achieved by replacing the estimate \hat{C}_{j+1} in (4.13) by $(1 + \epsilon)\hat{C}_{j+1}$. This is another commonly used inflation tactic; note, however, that it lacks the benefit of rank correction. It may therefore be combined with the additive inflation, yielding $\epsilon_1\hat{C} + (1 + \epsilon_2)\hat{C}_{j+1}$. More details regarding tuning of filters through inflation can be found in [4, 52, 77, 81, 50]. Another methodology that is important for practical implementation of the EnKF is **localization** [81, 50]. This is used to reduce unwanted correlations in \hat{C}_j between points that are separated by large distances in space. The underlying assumption is that the correla-

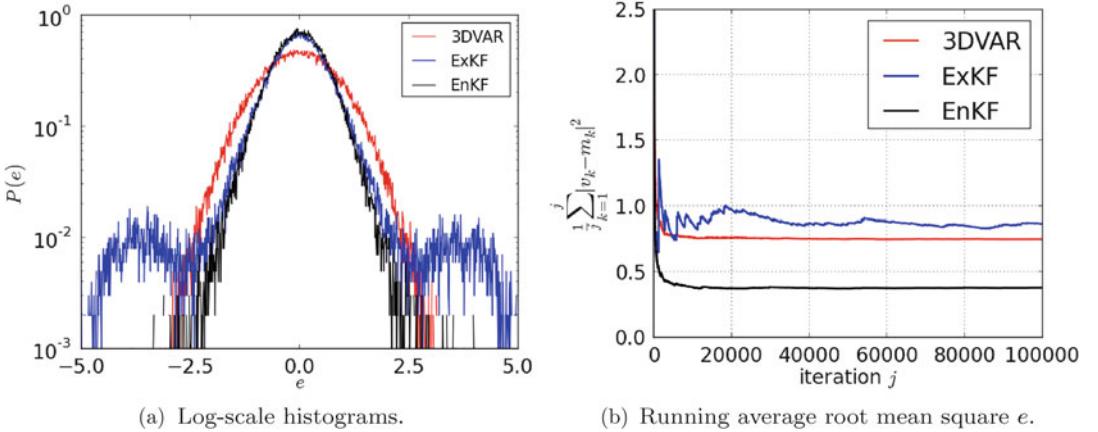


Fig. 4.11: Convergence of $e = m - v^\dagger$ for each filter for the sine map Example 2.3, corresponding to solutions from Figs. 4.5, 4.6, 4.7.

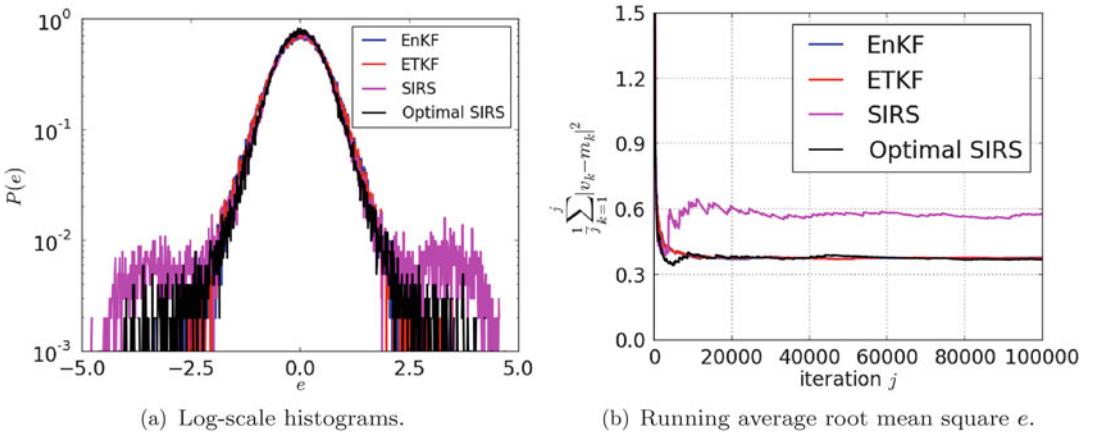


Fig. 4.12: Convergence of $e = m - v^\dagger$ for both versions of EnKF in comparison to the particle filters for the sine map Example 1.3, corresponding to solutions from Figs. 4.7, 4.8, 4.9, and 4.10.

tion between points decays proportionally to their distance from one another, and as such is increasingly corrupted by the sample error in ensemble methods. The sample covariance is hence modified to remove correlations between points separated by large distances in space. This is typically achieved by composing the empirical correlation matrix with a convolution kernel. Localization can have the further benefit of increasing rank, as in the case of the first type of variance inflation described above. An early reference illustrating the benefits and possible implementation of localization is [72]. An important reference that links this concept firmly with ideas from dynamical systems is [117].

Following the great success of the ensemble Kalman filter algorithm, in a series of papers [138, 20, 3, 146], the square-root filter framework was (re)discovered. The idea goes back at least to [5]. We focused the discussion above in Section 4.2.4 on the ETKF, but we note that it is possible to derive different transformations. For example, the singular evolutive interpolated Kalman (SEIK) filter proceeds by first projecting the ensemble into the

$(K - 1)$ -dimensional mean-free subspace, and then identifying a $(K - 1) \times (K - 1)$ matrix transformation, effectively prescribing a $K \times (K - 1)$ matrix transformation L_j as opposed to the $K \times K$ rank $(K - 1)$ matrix $T_j^{1/2}$ proposed in ETKF. The former is unique up to unitary transformation, while the latter is unique only up to unitary transformations that have $\mathbf{1}$ as an eigenvector. Other alternative transformations may take the form A_j or \tilde{K}_j such that $X_j = A_j \hat{X}_j$ or $X_j = (I - \tilde{K}H)\hat{X}_j$. These are known as the ensemble adjustment Kalman filter (EAKF) and the ensemble square-root filter (ESRF) respectively. See [20] for details about the ETKF, [3] for details about the EAKF, and [146] for details about the ESRF [146]. A review of all three is given in [138]. The similar singular evolutive interpolated Kalman (SEIK) filter was introduced in [121] and is compared with the other square root filters in [110]. Other ensemble-based filters have been developed in recent years bridging the ensemble Kalman filter with the particle filter, for example [71, 70, 123, 128].

- Section 4.3. In the linear case, the extended Kalman filter of course coincides with the Kalman filter; furthermore, in this case, the perturbed observation ensemble Kalman filter reproduces the true posterior distribution in the large-particle limit [50]. However, the filters introduced in Section 4.2 do not produce the correct posterior distribution when applied to general nonlinear problems. On the contrary, the particle filter *does* recover the true posterior distribution as the number of particles tends to infinity, as we show in Theorem 4.5. This proof is adapted from the very clear exposition in [122].

For more refined analyses of the convergence of particle filters, see, for example, [39, 46] and references therein. As explained in Remark 4.6 the constant appearing in the convergence results may depend exponentially on time if the mixing properties of the transition kernel $\mathbb{P}(dv_j|v_{j-1})$ are poor (the undesirable properties of deterministic dynamics illustrate this). There is also interesting work studying the effect of the dimension [129]. A proof that exploits ergodicity of the transition kernel, when that is present, may be found in [46]; the assumptions there on the transition and observation kernels are very strong and are generally not satisfied in practice, but studies indicate that comparable results may hold under less stringent conditions.

For a derivation and discussion of the optimal proposal, introduced in Section 4.3.3, see [47] and references therein. We also mention here the implicit filters developed by Chorin and coworkers [30, 29, 28]. These involve solving an implicit nonlinear equation for each particle that includes knowledge of the next set of observed data. This has some similarities to the method proposed in [143], and both are related to the optimal proposal mentioned above.

- Section 4.4. The stability of the Kalman filter is a well-studied subject, and the book [86] provides an excellent overview from the perspective of linear algebra. For extensions to the extended Kalman filter, see [77]. Theorem 4.10 provides a glimpse into the mechanisms at play within 3DVAR, and approximate Gaussian filters in general, in determining stability and accuracy: the incorporation of data can convert unstable dynamical systems, with positive Lyapunov exponents, into contractive nonautonomous dynamical systems, thereby leading, in the case of small observational noise, to filters that recover the true signal within a small error. This idea was highlighted in [27] and first studied rigorously for the 3DVAR method applied to the Navier–Stokes equation in [24]; this work was subsequently generalized to a variety of different models in [109, 88, 87]. It is also of note that these analyses of 3DVAR build heavily on ideas developed in [67] for a specialized form of data assimilation in which the observations are noise-free. In the language of the synchronization filter introduced in Section 4.4.3, this paper demonstrates the synchronization property (4.47) for the Navier–Stokes equation with sufficiently large number of Fourier mode observations, and for the Lorenz '63 model of Example 2.6 observed in only the first component. The

paper [87] consider similar issues for the Lorenz '96 model of Example 2.7. Similar ideas are studied for the perturbed observation EnKF in [82]; in this case, it is necessary to introduce a form of variance inflation to obtain a result analogous to Theorem 4.10. An important step in the theoretical analysis of ensemble Kalman filter methods is the paper [58], which uses ideas from the theory of shadowing in dynamical systems; the work proves that the ETKF variant can shadow the truth on arbitrarily long time intervals, provided the dimension of the ensemble is greater than the number of unstable directions in the system.

In the context of filter stability, it is important to understand the *optimality* of the mean of the true filtering distribution. We observe that all of the filtering algorithms that we have described produce an estimate of the probability distribution $\mathbb{P}(v_j|Y_j)$ that depends only on the data Y_j . There is a precise sense in which the true filtering distribution can be used to find a lower bound on the accuracy that can be achieved by any of these approximate algorithms. We let $\mathbb{E}(v_j|Y_j)$ denote the mean of v_j under the probability distribution $\mathbb{P}(v_j|Y_j)$ and let $E_j(Y_j)$ denote any estimate of the state v_j based only on data Y_j . Now consider all possible random data sets Y_j generated by the model (2.1), (2.2), noting that the randomness is generated by the initial condition v_0 and the noises $\{\xi_j, \eta_j\}$; in particular, conditioning on Y_j to obtain the probability distribution $\mathbb{P}(v_j|Y_j)$ can be thought of as being induced by conditioning on the observational noise $\{\eta_k\}_{k=1,\dots,j}$. Then $E_j^*(Y_j) := \mathbb{E}(v_j|Y_j)$ minimizes the mean-square error with respect to the random model (2.1), (2.2) [98, 77, 79]:

$$\mathbb{E}\|v_j - E_j^*(Y_j)\|^2 \leq \mathbb{E}\|v_j - E_j(Y_j)\|^2 \quad (4.49)$$

for all $E_j(Y_j)$. Thus the algorithms we have described can do no better at estimating the state of the system than can be achieved, in principle, from the conditional mean of the state given the data $\mathbb{E}(v_j|Y_j)$. This lower bound holds on average over all instances of the model. An alternative way to view the inequality (4.49) is as a means to providing upper bounds on the true filter. For example, under the conditions of Theorem 4.10, the right-hand side of (4.49) is, asymptotically as $j \rightarrow \infty$, of size $\mathcal{O}(\epsilon^2)$; thus we deduce that

$$\limsup_{j \rightarrow \infty} \mathbb{E}\|v_j - \mathbb{E}(v_j|Y_j)\|^2 \leq C\epsilon^2.$$

This viewpoint is adopted in [126], where the 3DVAR filter is used to bound the true filtering distribution. This latter optimality property can be viewed as resulting from the Galerkin orthogonality interpretation of the error resulting from taking conditional expectation.

We have considered large-time behavior on the assumption that the map Ψ can be implemented exactly. In situations in which the underlying map Ψ arises from a differential equation and numerical methods are required, large excursions in phase space caused by observational noise can cause numerical instabilities in the integration methods underlying the filters. Remark 8.7 illustrates this fact in the context of continuous time. See [59] for a discussion of this issue.

- Section 4.5. We mention here the *rank histogram*. This is another consistency check on the output of ensemble- or particle-based approximations of the filtering distribution. The idea is to consider observed scalar quantities and to generate ordered bins associated to that scalar and then keep track of the statistics over time of the data y_j with respect to the bins. For example, if one has an approximation of the distribution consisting of N equally weighted particles, then a rank histogram for the first component of the state consists of three steps, each carried out at each time j . First, add a random draw from the observational noise $N(0, \Gamma)$ to each particle after the prediction phase of the algorithm.

Second, order the particles according to the value of their first component, generating $N-1$ bins between the values of the first component of each particle, and with one extra bin on each end. Finally, rank the current observation y_j between 1 and $N+1$ depending on which bin it lands in. If one does this at each time j , a histogram of the rank of the observations is obtained. The “spread” of the ensemble can be evaluated using this diagnostic. If the histogram is uniform, then the spread is consistent. If it is concentrated toward the center, then the spread is overestimated. If it is concentrated at the edges, then the spread is underestimated. This consistency check on the statistical model used was introduced in [2] and is widely adopted throughout the data-assimilation community.

4.7 Exercises

1. Consider the Kalman filter in the case $M = H = I$, $\Sigma = 0$, and $\Gamma > 0$. Prove that the covariance operator C_j converges to 0 as $j \rightarrow \infty$. Modify the program p8.m so that it applies to this setup, in the two-dimensional setting. Verify what you have proved regarding the covariance and make a conjecture about the limiting behavior of the mean of the Kalman filter.
2. Consider the 3DVAR algorithm in the case $\Psi(v) = v$, $H = I$, $\Sigma = 0$, and $\Gamma > 0$. Choose $\hat{C} = \alpha\Gamma$. Find an equation for the error $e_j := v_j - m_j$ and derive an expression for $\lim_{j \rightarrow \infty} \mathbb{E}|e_j|^2$ in terms of α and $\sigma^2 := \mathbb{E}|\eta_j|^2$. Modify the program p9.m so that it applies to this setup, in the one-dimensional setting. Verify what you have proved regarding the limiting behavior of the m_j .
3. Consider the EnKF algorithm in the same setting as the previous example. Modify program p12.m so that it applies to this setup, in the one-dimensional setting. Study the behavior of the sequence m_j found as the mean of the particles $v_j^{(n)}$ over the ensemble index n .
4. Consider the SIRS algorithm in the same setting as the previous example. Modify program p14.m so that it applies to this setup, in the one-dimensional setting. Study the behavior of the sequence m_j found as the mean of the particles $v_j^{(n)}$ over the ensemble index n .
5. Make comparative comments regarding the 3DVAR, EnKF, and SIRS algorithms on the basis of your solutions to the three preceding exercises.
6. In this exercise, we study the behavior of the mean of the Kalman filter in the case of one-dimensional dynamics. The notation follows the development in Section 4.4.1. Consider the case $\sigma = 0$ and assume that the data $\{y_j\}_{j \in \mathbb{N}}$ is generated from a true signal $\{v_j^\dagger\}_{j \in \mathbb{Z}^+}$ governed by the equation

$$v_{j+1}^\dagger = \lambda v_j^\dagger,$$

and that the additive observational noise $\{\eta_j\}_{j \in \mathbb{N}}$ is drawn from an i.i.d. sequence with variance γ^2 . Define the error $e_j = m_j - v_j^\dagger$ between the estimated mean and the true signal and use (4.37b) to show that

$$e_{j+1} = \left(1 - \frac{c_{j+1}}{\gamma^2}\right) \lambda e_j + \frac{c_{j+1}}{\gamma^2} \eta_{j+1}. \quad (4.50)$$

Deduce that e_j is Gaussian and that its mean and covariance satisfy the equations

$$\mathbb{E}e_{j+1} = \lambda \left(1 - \frac{c_{j+1}}{\gamma^2}\right) \mathbb{E}e_j \quad (4.51)$$

and

$$\mathbb{E}e_{j+1}^2 = \lambda^2 \left(1 - \frac{c_{j+1}}{\gamma^2}\right)^2 \mathbb{E}e_j^2 + \frac{c_{j+1}^2}{\gamma^2}. \quad (4.52)$$

Equation (4.51) can be solved to obtain

$$\mathbb{E}e_j = \lambda^j \left[\prod_{i=0}^j \left(1 - \frac{c_{i+1}}{\gamma^2}\right) \right] \mathbb{E}e_0. \quad (4.53)$$

In a similar way, obtain for the solution of (4.52),

$$\mathbb{E}e_j^2 = \lambda^{2j} \left[\prod_{i=0}^{j-1} \left(1 - \frac{c_{i+1}}{\gamma^2}\right)^2 \right] \mathbb{E}e_0^2 + \sum_{i=0}^{j-1} \left\{ \left[\prod_{k=i+1}^j \left(1 - \frac{c_k}{\gamma^2}\right) \right] \lambda^{2(j-i)} \frac{c_i^2}{\gamma^2} \right\} + \frac{c_j^2}{\gamma^2}. \quad (4.54)$$

Using the properties of the variance derived in Section 4.4.1, prove that the mean of the error tends to zero and that the asymptotic variance is bounded by γ^2 .

Chapter 5

Discrete Time: MATLAB Programs

This chapter is dedicated to illustrating the examples, theory, and algorithms, as presented in the previous chapters, through a few short and easy-to-follow MATLAB programs. These programs are provided for two reasons: (i) For some readers, they will form the best route by which to appreciate the details of the examples, theory, and algorithms we describe; (ii) for other readers, they will be a useful starting point to develop their own codes. While ours are not necessarily the optimal implementations of the algorithms discussed in these notes, they have been structured to be simple to understand, to modify, and to extend. In particular, the code may be readily extended to solve problems more complex than those described in Examples 2.1–2.7, which we will use for most of our illustrations. The chapter is divided into three sections, corresponding to programs relevant to each of the preceding three chapters.

Before getting into details, we highlight a few principles that have been adopted in the programs and in the accompanying text of this chapter. First, notation is consistent between programs, and it matches the text in the previous sections of the book as far as possible. Second, since many of the elements of the individual programs are repeated, they will be described in detail only in the text corresponding to the program in which they first appear; the short annotations explaining them will, however, be repeated within the programs. Third, the reader is advised to use the documentation available at the command line for *any* built-in functions of MATLAB; this information can be accessed using the `help` command—for example, the documentation for the command `help` can be accessed by typing `help help`.

5.1 Chapter 2 Programs

The programs `p1.m` and `p2.m` used to generate the figures in Chapter 2 are presented in this section. Thus these algorithms simply solve the dynamical system (2.1) and process the resulting data.

Electronic supplementary material The online version of this chapter (doi: [10.1007/978-3-319-20325-6_5](https://doi.org/10.1007/978-3-319-20325-6_5)) contains supplementary material, which is available to authorized users.

5.1.1. p1.m

The first program, `p1.m`, illustrates how to obtain sample paths from equations (2.1) and (2.3). In particular, the program simulates sample paths of the equation

$$u_{j+1} = \alpha \sin(u_j) + \xi_j, \quad (5.1)$$

with $\xi_j \sim N(0, \sigma^2)$ i.i.d. and $\alpha = 2.5$, both for deterministic ($\sigma = 0$) and stochastic dynamics ($\sigma \neq 0$) corresponding to Example 2.3. In line 5, the variable `J` is defined, which corresponds to the number of forward steps that we will take. The parameters α and σ are set in lines 6–7. The seed for the random-number generator is set to `sd ∈ N` in line 8 using the command `rng(sd)`. This guarantees that the results will be reproduced exactly by running the program with this same `sd`. Different choices of `sd ∈ N` will lead to different streams of random numbers used in the program, which may also be desirable in order to observe the effects of different random numbers on the output. The command `sd` will be called in the preamble of all of the programs that follow. In line 9, two vectors of length `J` are created, named `v` and `vnoise`; after the program has run, these two vectors contain the solutions for the case of deterministic ($\sigma = 0$) and stochastic dynamics ($\sigma = 0.25$) respectively. After the initial conditions are set in line 10, the desired map is iterated, without and with noise, in lines 12–15. Note that the only difference between the forward iteration of `v` and that of `vnoise` is the presence of the `sigma*randn` term, which corresponds to the generation of a random variable sampled from $N(0, \sigma^2)$. Lines 17–18 contain code that graphs the trajectories, with and without noise, to produce Figure 2.3. Figures 2.1, 2.2, and 2.5 were obtained by simply modifying lines 12–15 of this program, in order to create sample paths for the corresponding Ψ for the other three examples; furthermore, Figure 2.4a was generated from output of this program, and Figure 2.4b was generated from output of a modification of this program.

```

1 clear;set(0,'defaultaxesfontsize',20);format long
2 %% p1.m - behaviour of sin map (Ex. 1.3)
3 %% with and without observational noise
4
5 J=10000;% number of steps
6 alpha=2.5;% dynamics determined by alpha
7 sigma=0.25;% dynamics noise variance is sigma^2
8 sd=1;rng(sd);% choose random number seed
9 v=zeros(J+1,1); vnoise=zeros(J+1,1);% preallocate space
10 v(1)=1;vnoise(1)=1;% initial conditions
11
12 for i=1:J
13     v(i+1)=alpha*sin(v(i));
14     vnoise(i+1)=alpha*sin(vnoise(i))+sigma*randn;
15 end
16
17 figure(1), plot([0:1:J],v),
18 figure(2), plot([0:1:J],vnoise),

```

5.1.2. p2.m

The second program presented here, `p2.m`, is designed to visualize the posterior distribution in the case of one-dimensional deterministic dynamics. For clarity, the program is separated into three main sections. The `setup` section in lines 5–10 defines the parameters of the problem. The model parameter `r` is defined in line 6, and it determines the dynamics of the forward model, in this case given by the logistic map (2.9):

$$v_{j+1} = rv_j(1 - v_j). \quad (5.2)$$

The dynamics are taken as deterministic, so the parameter `sigma` does not feature here. The parameter `r` is equal to 2, so that the dynamics are not chaotic, as the explicit solution given in Example 2.4 shows. The parameters `m0` and `C0` define the mean and covariance of the prior distribution $v_0 \sim N(m_0, C_0)$, while `gamma` defines the observational noise $\eta_j \sim N(0, \gamma^2)$.

The `truth` section in lines 14–20 generates the true reference trajectory (or truth) `vt` in line 18 given by (5.2), as well as the observations `y` in line 19 given by

$$y_j = v_j + \eta_j. \quad (5.3)$$

Note that the index of `y(:, j)` corresponds to observation of `H*v(:, j+1)`. This is due to the fact that the first index of an array in MATLAB is `j=1`, while the initial condition is v_0 , and the first observation is of v_1 . So effectively the indices of `y` are correct as corresponding to the text and equation (5.3), but the indices of `v` are off by 1. The memory for these vectors is *preallocated* in line 14. This is unnecessary, because MATLAB would simply *dynamically allocate* the memory in its absence, but it would slow down the computations due to the necessity of allocating new memory each time the given array changes size. Commenting this line out allows observation of this effect, which becomes significant when `J` becomes sufficiently large.

The `solution` section after line 24 computes the solution, in this case the pointwise representation of the posterior smoothing distribution on the scalar initial condition. The pointwise values of the initial condition are given by the vector `v0` (v_0) defined in line 24. There are many ways to construct such vectors, and this convention defines the initial (0.01) and final (0.99) values and a uniform step size 0.0005. It is also possible to use the command `v0=linspace(0.01, 0.99, 1961)`, defining the *number* 1961 of intermediate points, rather than the step size 0.0005. The corresponding vector of values of `Phidet` (Φ_{det}), `Jdet` (J_{det}), and `Idet` (I_{det}) are computed in lines 32, 29, and 34 for each value of `v0`, as related by the equation

$$I_{\text{det}}(v_0; y) = J_{\text{det}}(v_0) + \Phi_{\text{det}}(v_0; y), \quad (5.4)$$

where $J_{\text{det}}(v_0)$ is the **background** penalization and $\Phi_{\text{det}}(v_0; y)$ is the **model–data misfit** functional given by (2.29b) and (2.29c) respectively. The function $I_{\text{det}}(v_0; y)$ is the negative log-posterior as given in Theorem 2.11. Having obtained $I_{\text{det}}(v_0; y)$, we calculate $\mathbb{P}(v_0|y)$ in lines 37–38, using the formula

$$\mathbb{P}(v_0|y) = \frac{\exp(-I_{\text{det}}(v_0; y))}{\int \exp(-I_{\text{det}}(v_0; y))}. \quad (5.5)$$

The trajectory `v` corresponding to the given value of `v0` (`v0(i)`) is denoted by `vv` and is replaced for each new value of `v0(i)` in lines 29 and 31, since it is required only to compute `Idet`. The command `trapz(v0, exp(-Idet))` in line 37 approximates the denominator of the above by the trapezoidal rule, i.e., the summation

$$\text{trapz}(v0, \exp(-Idet)) = \sum_{i=1}^{N-1} (v0(i+1) - v0(i)) * (Idet(i+1) + Idet(i))/2. \quad (5.6)$$

The rest of the program deals with plotting our results, and in this instance, it coincides with the output of Figure 2.11b. Again, simple modifications of this program were used to produce Figures 2.10, 2.12, and 2.13. Note that `rng(sd)` in line 8 allows us to use the same random numbers every time the file is executed; those random numbers are generated with the seed `sd`, as described in Section 5.1.1. Commenting this line out would result in the creation of new realizations of the random data y , different from those used to obtain Figure 2.11b.

```

1 clear; set(0,'defaultaxesfontsize',20); format long
2 %% p2.m smoothing problem for the deterministic logistic map (Ex. 1.4)
3 %% setup
4
5 J=1000;% number of steps
6 r=2;% dynamics determined by r
7 gamma=0.1;% observational noise variance is gamma^2
8 C0=0.01;% prior initial condition variance
9 m0=0.7;% prior initial condition mean
10 sd=1;rng(sd);% choose random number seed
11
12 %% truth
13
14 vt=zeros(J+1,1); y=zeros(J,1);% preallocate space to save time
15 vt(1)=0.1;% truth initial condition
16 for j=1:J
17     % can be replaced by Psi for each problem
18     vt(j+1)=r*vt(j)*(1-vt(j));% create truth
19     y(j)=vt(j+1)+gamma*randn;% create data
20 end
21
22 %% solution
23
24 v0=[0.01:0.0005:0.99];% construct vector of different initial data
25 Phidet=zeros(length(v0),1);Idet=Phidet;Jdet=Phidet;% preallocate space
26 vv=zeros(J,1);% preallocate space
27 % loop through initial conditions vv0, and compute log posterior I0(vv0)
28 for j=1:length(v0)
29     vv(1)=v0(j); Jdet(j)=1/2/C0*(v0(j)-m0)^2;% background penalization
30     for i=1:J
31         vv(i+1)=r*vv(i)*(1-vv(i));
32         Phidet(j)=Phidet(j)+1/2/gamma^2*(y(i)-vv(i+1))^2;% misfit
33             functional
34     end
35     Idet(j)=Phidet(j)+Jdet(j);
36 end
37
38 constant=trapz(v0,exp(-Idet));% approximate normalizing constant
39 P=exp(-Idet)/constant;% normalize posterior distribution
40 prior=normpdf(v0,m0,sqrt(C0));% calculate prior distribution
41
42 figure(1),plot(v0,prior,'k','LineWidth',2)
43 hold on, plot(v0,P,'r--','LineWidth',2), xlabel 'v_0',
44 legend 'prior' J=10^3

```

5.2 Chapter 3 Programs

The programs `p3.m-p7.m`, used to generate the figures in Chapter 3, are presented in this section. Hence various MCMC algorithms used to sample the posterior smoothing distribution are given. Furthermore, optimization algorithms used to obtain solutions of the 4DVAR and w4DVAR variational methods are also introduced. Our general theoretical development of MCMC methods in Section 3.2 employs a notation of u for the state of the chain and w for the proposal. For deterministic dynamics, the state is the initial condition v_0 ; for stochastic dynamics, it is either the signal v or the pair (v_0, ξ) , where ξ is the noise (since this pair determines the signal). Where appropriate, the programs described here use the letter v , and variants thereof, for the state of the Markov chain to keep the connection with the underlying dynamics model.

5.2.1. p3.m

The program `p3.m` contains an implementation of the random walk Metropolis (RWM) MCMC algorithm. The development follows Section 3.2.3, where the algorithm is used to determine the posterior distribution on the initial condition arising from the deterministic logistic map of Example 2.4 given by (5.2). Note that in this case, since the underlying dynamics are deterministic and hence completely determined by the initial condition, the RWM algorithm will provide samples from a probability distribution on \mathbb{R} .

As in program `p2.m`, the code is divided into three sections: `setup`, where parameters are defined; `truth`, where the truth and data are generated; and `solution`, where the solution is computed, this time by means of MCMC samples from the posterior smoothing distribution. The parameters in lines 5–10 and the true solution (here taken as only the initial condition, rather than the trajectory it gives rise to) `vt` in line 14 are taken to be the same as those used to generate Figure 2.13. The temporary vector `vv` generated in line 19 is the trajectory corresponding to the truth (`vv(1)=vt` in line 14) and used to calculate the observations `y` in line 20. The true value `vt` will also be used as the initial sample in the Markov chain for this and for all subsequent MCMC programs. This scenario is, of course, impossible in the case that the data is not simulated. However, it is useful when the data is simulated, as it is here, because it can reduce the burn-in time, i.e., the time necessary for the current sample in the chain to reach the target distribution, or the high-probability region of the state space. Because we initialize the Markov chain at the truth, the value of $I_{\text{det}}(v^\dagger)$, denoted by the temporary variable `Idet`, is required to determine the initial acceptance probability, as described below. It is computed in lines 15–23 exactly as in lines 25–34 of program `p2.m`, as described around equation (5.4).

In the `solution` section, some additional MCMC parameters are defined. In line 28, the number of samples is set to `N = 10^5`. For the parameters and specific data used here, this is sufficient for convergence of the Markov chain. In line 30, the step-size parameter `beta` is preset such that the algorithm for this particular posterior distribution has a reasonable acceptance probability, or ratio of accepted to rejected moves. A general rule of thumb for this is that it should be somewhere around 0.5, to ensure that the algorithm is not too correlated because of high rejection rate (acceptance probability near zero) and that it is not too correlated because of small moves (acceptance probability near one). The vector `V` defined in line 29 will save all of the samples. This is an example in which preallocation is *very* important. Try using the commands `tic` and `toc` before and respectively after the loop

in lines 33–50 in order to time the chain both with and without preallocation.¹ In line 34, a move is proposed according to the proposal equation (3.15):

$$w^{(k)} = v^{(k-1)} + \beta \iota^{(k-1)},$$

where $v(v)$ is the current state of the chain (initially taken to be equal to the true initial condition v_0), $\iota^{(k-1)}=\text{randn}$ is an i.i.d. standard normal, and w represents $w^{(k)}$. Indices are not used for v and w because they will be overwritten at each iteration.

The temporary variable vv is again used for the trajectory corresponding to $w^{(k)}$ as a vehicle to compute the value of the proposed $l_{\text{det}}(w^{(k)}; y)$, denoted in line 42 by $I0prop = J0prop + PhiProp$. In lines 44–46, the decision to accept or reject the proposal is made based on the acceptance probability

$$a(v^{(k-1)}, w^{(k)}) = 1 \wedge \exp(l_{\text{det}}(v^{(k-1)}; y) - l_{\text{det}}(w^{(k)}; y)).$$

In practice, this corresponds to drawing a uniform random number rand and replacing v and I_{det} in line 45 with w and $I0prop$ if $\text{rand} < \exp(I0 - I0prop)$ in line 44. The variable bb is incremented if the proposal is accepted, so that the running ratio of accepted moves bb to total steps n can be computed in line 47. This approximates the average acceptance probability. The current sample $v^{(k)}$ is stored in line 48. Notice that here one could replace v by $V(n-1)$ in line 34, and by $V(n)$ in line 45, thereby eliminating v , and letting w be the only temporary variable. However, the present construction is favorable, because as mentioned above, in general one may not wish to save every sample.

The samples V are used in lines 51–53 to visualize the posterior distribution. In particular, bins of width dx are defined in line 51, and the command `hist` is used in line 52. The assignment $Z = \text{hist}(V, v0)$ means first that the real number line is split into M bins with centers defined according to $v0(i)$ for $i = 1, \dots, M$, with the first and last bins corresponding to the negative, respectively positive, half-lines. Second, $Z(i)$ counts the number of k for which $V(k)$ is in the bin with center determined by $v0(i)$. Again, `trapz` (5.6) is used to compute the normalizing constant in line 53, directly within the plotting command. The choice of the location of the histogram bins allows for a direct comparison with the posterior distribution calculated from the program `p2.m` by directly evaluating $l_{\text{det}}(v; y)$ defined in (5.4) for different values of initial conditions v . This output is then compared with the corresponding output of `p2.m` for the same parameters in Figure 3.2.

¹ In practice, one may often choose to collect certain statistics from the chain “on the fly” rather than saving every sample, particularly if the state space is high-dimensional and the memory required for each sample is large.

```

1 clear; set(0,'defaultaxesfontsize',20); format long
2 %% p3.m MCMC RWM algorithm for logistic map (Ex. 1.4)
3 %% setup
4
5 J=5;% number of steps
6 r=4;% dynamics determined by alpha
7 gamma=0.2;% observational noise variance is gamma^2
8 C0=0.01;% prior initial condition variance
9 m0=0.5;% prior initial condition mean
10 sd=10;rng(sd);% choose random number seed
11
12 %% truth
13
14 vt=0.3;vv(1)=vt;% truth initial condition
15 Jdet=1/2/C0*(vt-m0)^2;% background penalization
16 Phidet=0;% initialization model-data misfit functional
17 for j=1:J
18     % can be replaced by Psi for each problem
19     vv(j+1)=r*vv(j)*(1-vv(j));% create truth
20     y(j)=vv(j+1)+gamma*randn;% create data
21     Phidet=Phidet+1/2/gamma^2*(y(j)-vv(j+1))^2;% misfit functional
22 end
23 Idet=Jdet+Phidet;% compute log posterior of the truth
24
25 %% solution
26 % Markov Chain Monte Carlo: N forward steps of the
27 % Markov Chain on R (with truth initial condition)
28 N=1e5;% number of samples
29 V=zeros(N,1);% preallocate space to save time
30 beta=0.05;% step-size of random walker
31 v=vt;% truth initial condition (or else update I0)
32 n=1; bb=0; rat(1)=0;
33 while n<=N
34     w=v+sqrt(2*beta)*randn;% propose sample from random walker
35     vv(1)=w;
36     Jdetprop=1/2/C0*(w-m0)^2;% background penalization
37     Phidetprop=0;
38     for i=1:J
39         vv(i+1)=r*vv(i)*(1-vv(i));
40         Phidetprop=Phidetprop+1/2/gamma^2*(y(i)-vv(i+1))^2;
41     end
42     Idetprop=Jdetprop+Phidetprop;% compute log posterior of the proposal
43
44     if rand<exp(Idet-Idetprop)% accept or reject proposed sample
45         v=w; Idet=Idetprop; bb=bb+1;% update the Markov chain
46     end
47     rat(n)=bb/n;% running rate of acceptance
48     V(n)=v;% store the chain
49     n=n+1;
50 end
51 dx=0.0005; v0=[0.01:dx:0.99];
52 Z=hist(V,v0);% construct the posterior histogram
53 figure(1), plot(v0,Z/trapz(v0,Z),'k','Linewidth',2)% visualize the
54 posterior

```

5.2.2. p4.m

The program `p4.m` contains an implementation of the independence dynamics sampler for stochastic dynamics, as introduced in Section 3.2.4. Thus the posterior distribution is on the entire signal $\{v_j\}_{j \in \mathbb{J}}$. The forward model in this case is from Example 2.3, given by (5.1). The smoothing distribution $\mathbb{P}(v|Y)$ is therefore over the state space \mathbb{R}^{J+1} .

The sections `setup`, `truth`, and `solution` are defined as for program `p3.m`, but note that now the smoothing distribution is over the entire path, not just over the initial condition, because we are considering stochastic dynamics. Since the state space is now the path space, rather than the initial condition as it was in program `p3.m`, the truth $v_t \in \mathbb{R}^{J+1}$ is now a vector. Its initial condition is taken as a draw from $N(m_0, C_0)$ in line 16, and the trajectory is computed in line 20, so that at the end, $v_t \sim \rho_0$. As in program `p3.m`, $v^\dagger(v_t)$ will be the chosen initial condition in the Markov chain (to ameliorate burn-in issues), and so $\Phi(v^\dagger; y)$ is computed in line 23. Recall from Section 3.2.4 that only $\Phi(\cdot; y)$ is required to compute the acceptance probability in this algorithm.

Notice that the collection of samples $v \in \mathbb{R}^{N \times J+1}$ preallocated in line 30 is substantial in this case, illustrating the memory issue that arises when the dimension of the signal space, and number of samples, increases.

The current state of the chain $v^{(k)}$ and the value of $\Phi(v^{(k)}; y)$ are again denoted by v and Phi , while the proposal $w^{(k)}$ and the value of $\Phi(w^{(k)}; y)$ are again denoted by w and PhiProp , as in program `p3`. As discussed in Section 3.2.4, the proposal $w^{(k)}$ is an independent sample from the prior distribution ρ_0 , similarly to v^\dagger , and it is constructed in lines 34–39. The acceptance probability used in line 40 is now

$$a(v^{(k-1)}, w^{(k)}) = 1 \wedge \exp(\Phi(v^{(k-1)}; y) - \Phi(w^{(k)}; y)). \quad (5.7)$$

The remainder of the program is structurally the same as `p3.m`. The outputs of this program are used to plot Figures 3.3, 3.4, and 3.5. Note that in the case of Figure 3.5, we have used $N = 10^8$ samples.

```

1 clear; set(0,'defaultaxesfontsize',20); format long
2 %% p4.m MCMC INDEPENDENCE DYNAMICS SAMPLER algorithm
3 %% for sin map (Ex. 1.3) with noise
4 %% setup
5
6 J=10;% number of steps
7 alpha=2.5;% dynamics determined by alpha
8 gamma=1;% observational noise variance is gamma^2
9 sigma=1;% dynamics noise variance is sigma^2
10 C0=1;% prior initial condition variance
11 m0=0;% prior initial condition mean
12 sd=0;rng(sd);% choose random number seed
13
14 %% truth
15
16 vt(1)=m0+sqrt(C0)*randn;% truth initial condition
17 Phi=0;
18
19 for j=1:J
20     vt(j+1)=alpha*sin(vt(j))+sigma*randn;% create truth
21     y(j)=vt(j+1)+gamma*randn;% create data
22     % calculate log likelihood of truth, Phi(v;y) from (1.11)
23     Phi=Phi+1/2/gamma^2*(y(j)-vt(j+1))^2;
24 end
25
26 %% solution
27 % Markov Chain Monte Carlo: N forward steps of the
28 % Markov Chain on R^{J+1} with truth initial condition
29 N=1e5;% number of samples
30 V=zeros(N,J+1);% preallocate space to save time
31 v=vt;% truth initial condition (or else update Phi)
32 n=1; bb=0; rat(1)=0;
33 while n<=N
34     w(1)=sqrt(C0)*randn;% propose sample from the prior
35     Phiprop=0;
36     for j=1:J
37         w(j+1)=alpha*sin(w(j))+sigma*randn;% propose sample from the prior
38         Phiprop=Phiprop+1/2/gamma^2*(y(j)-w(j+1))^2;% compute likelihood
39     end
40     if rand<exp(Phi-Phiprop)% accept or reject proposed sample
41         v=w; Phi=Phiprop; bb=bb+1;% update the Markov chain
42     end
43     rat(n)=bb/n;% running rate of acceptance
44     V(n,:)=v;% store the chain
45     n=n+1;
46 end
47 % plot acceptance ratio and cumulative sample mean
48 figure;plot(rat)
49 figure;plot(cumsum(V(1:N,1))./[1:N]')
50 xlabel('samples N')
51 ylabel('(1/N) \Sigma_{n=1}^N v_0^{(n)}')

```

5.2.3. p5.m

The independence dynamics sampler of Section 5.2.2 may be very inefficient, since typical random draws from the dynamics may be unlikely to fit the data as well as the current state, and will then be rejected. The fifth program, p5.m, gives an implementation of the pCN algorithm from Section 3.2.4 that is designed to overcome this issue by including the parameter β , which, if chosen small, allows for incremental steps in signal space and hence the possibility of nonnegligible acceptance probabilities. This program is used to generate Figure 3.6

This program is almost identical to p4.m, and so only the points at which it differs will be described. First, since the acceptance probability is given by

$$a(v^{(k-1)}, w^{(k)}) = 1 \wedge \exp(\Phi(v^{(k-1)}; y) - \Phi(w^{(k)}; y) + G(v^{(k-1)}) - G(w^{(k)})),$$

the quantity

$$G(u) = \sum_{j=0}^{J-1} \left(\frac{1}{2} |\Sigma^{-\frac{1}{2}} \Psi(u_j)|^2 - \langle \Sigma^{-\frac{1}{2}} u_{j+1}, \Sigma^{-\frac{1}{2}} \Psi(u_j) \rangle \right)$$

will need to be computed, both for $v^{(k)}$ (denoted by v in lines 31 and 44), where its value is denoted by G ($v^{(0)} = v^\dagger$), as well as for $G(v^\dagger)$ which is computed in line 22), and for $w^{(k)}$ (denoted by w in line 36) where its value is denoted by Gprop in line 39.

As discussed in Section 3.2.4, the proposal $w^{(k)}$ is given by (3.19):

$$w^{(k)} = m + (1 - \beta^2)^{\frac{1}{2}} (v^{(k-1)} - m) + \beta \iota^{(k-1)}; \quad (5.8)$$

here $\iota^{(k-1)} \sim N(0, C)$ are i.i.d. and denoted by iota in line 35. Here C is the covariance of the Gaussian measure π_0 given in Equation (2.24) corresponding to the case of trivial dynamics $\Psi = 0$, and m is the mean of π_0 . The value of m is given by m in line 33.

```

1 clear;set(0,'defaultaxesfontsize',20);format long
2 %% p5.m MCMC pCN algorithm for sin map (Ex. 1.3) with noise
3
4 %% setup
5 J=10;% number of steps
6 alpha=2.5;% dynamics determined by alpha
7 gamma=1;% observational noise variance is gamma^2
8 sigma=.1;% dynamics noise variance is sigma^2
9 C0=1;% prior initial condition variance
10 m0=0;% prior initial condition mean
11 sd=0;rng(sd);% Choose random number seed
12
13 %% truth
14
15 vt(1)=m0+sqrt(C0)*randn;% truth initial condition
16 G=0;Phi=0;
17
18 for j=1:J
19     vt(j+1)=alpha*sin(vt(j))+sigma*randn;% create truth
20     y(j)=vt(j+1)+gamma*randn;% create data
21     % calculate log density from (1--)
22     G=G+1/2/sigma^2*((alpha*sin(vt(j)))^2-2*vt(j+1)*alpha*sin(vt(j)));
23     % calculate log likelihood phi(u;y) from (1.11)
24     Phi=Phi+1/2/gamma^2*(y(j)-vt(j+1))^2;
25 end
26
27 %% solution
28 % Markov Chain Monte Carlo: N forward steps
29 N=1e5;% number of samples
30 beta=0.02;% step-size of pCN walker
31 v=vt;% truth initial condition (or update G + Phi)
32 V=zeros(N,J+1); n=1; bb=0; rat=0;
33 m=[m0,zeros(1,J)];
34 while n<=N
35     iota=[sqrt(C0)*randn,sigma*randn(1,J)];% Gaussian prior sample
36     w=m+sqrt(1-beta^2)*(v-m)+beta*iota;% propose sample from the pCN walker
37     Gprop=0;Phiprop=0;
38     for j=1:J
39         Gprop=Gprop+1/2/sigma^2*((alpha*sin(w(j)))^2-2*w(j+1)*alpha*sin
40             (w(j)));
41         Phiprop=Phiprop+1/2/gamma^2*(y(j)-w(j+1))^2;
42     end
43     if rand<exp(Phi-Phiprop+G-Gprop)% accept or reject proposed sample
44         v=w;Phi=Phiprop;G=Gprop;bb=bb+1;% update the Markov chain
45     end
46     rat(n)=bb/n;% running rate of acceptance
47     V(n,:)=v;% store the chain
48     n=n+1;
49 end
50 % plot acceptance ratio and cumulative sample mean
51 figure;plot(rat)
52 figure;plot(cumsum(V(1:N,1))./[1:N]')
53 xlabel('samples N')
54 ylabel('(1/N) \Sigma_{n=1}^N v_0^{(n)}')

```

5.2.4. p6.m

The pCN dynamics sampler is now introduced as program `p6.m`. The independence dynamics sampler of Section 5.2.2 may be viewed as a special case of this algorithm for proposal variance $\beta = 1$. This proposal combines the benefits of tuning the step size β while still respecting the prior distribution on the dynamics. It does so by sampling the initial condition and noise (v_0, ξ) rather than the path itself, in lines 34 and 35, as given by equation (5.8). However, as opposed to the pCN sampler of the previous section, this variable `w` is now interpreted as a sample of (v_0, ξ) and is therefore fed into the path `vv` itself in line 39. The acceptance probability is the same as that of the independence dynamics sampler (5.7), depending only on `Phi`. If the proposal is accepted, both the forcing `u=w` and the path `v=vv` are updated in line 44. Only the path is saved, as in the previous routines, in line 47.

```

1 clear;set(0,'defaultaxesfontsize',20);format long
2 %% p6.m MCMC pCN Dynamics algorithm for
3 %% sin map (Ex. 1.3) with noise
4 %% setup
5
6 J=10;% number of steps
7 alpha=2.5;% dynamics determined by alpha
8 gamma=1;% observational noise variance is gamma^2
9 sigma=1;% dynamics noise variance is sigma^2
10 C0=1;% prior initial condition variance
11 m0=0;% prior initial condition mean
12 sd=0;rng(sd);% Choose random number seed
13
14 %% truth
15
16 vt(1)=m0+sqrt(C0)*randn;% truth initial condition
17 ut(1)=vt(1);
18 Phi=0;
19 for j=1:J
20     ut(j+1)=sigma*randn;
21     vt(j+1)=alpha*sin(vt(j))+ut(j+1);% create truth
22     y(j)=vt(j+1)+gamma*randn;% create data
23     % calculate log likelihood phi(u;y) from (1.11)
24     Phi=Phi+1/2/gamma^2*(y(j)-vt(j+1))^2;
25 end
26
27 %% solution
28 % Markov Chain Monte Carlo: N forward steps
29 N=1e5;% number of samples
30 beta=0.2;% step-size of pCN walker
31 u=ut;v=vt;% truth initial condition (or update Phi)
32 V=zeros(N,J+1); n=1; bb=0; rat=0;m=[m0,zeros(1,J)];
33 while n<=N
34     iota=[sqrt(C0)*randn,sigma*randn(1,J)];% Gaussian prior sample
35     w=m+sqrt(1-beta^2)*(u-m)+beta*iota;% propose sample from the pCN walker
36     vv(1)=w(1);
37     Phiprop=0;
38     for j=1:J
39         vv(j+1)=alpha*sin(vv(j))+w(j+1);% create path
40         Phiprop=Phiprop+1/2/gamma^2*(y(j)-vv(j+1))^2;
41     end
42
43     if rand<exp(Phi-Phiprop)% accept or reject proposed sample
44         u=w;v=vv;Phi=Phiprop;bb=bb+1;% update the Markov chain
45     end
46     rat(n)=bb/n;% running rate of acceptance
47     V(n,:)=v;% store the chain
48     n=n+1;
49 end
50 % plot acceptance ratio and cumulative sample mean
51 figure;plot(rat)
52 figure;plot(cumsum(V(1:N,1))./[1:N]')
53 xlabel('samples N')
54 ylabel('(1/N) \Sigma_{n=1}^N v_0^{(n)}')

```

5.2.5. p7.m

The next program, `p7.m`, contains an implementation of the weak constrained variational algorithm w4DVAR discussed in Section 3.3. This program is written as a function, while all previous programs were written as scripts. This choice was made for `p7.m` so that the MATLAB built-in function `fminsearch` can be used for optimization in the `solution` section, and the program can still be self-contained. To use this built-in function, it is necessary to define an *auxiliary* objective function \mathbf{I} to be optimized. The function `fminsearch` can be used within a script, but the auxiliary function would then have to be written separately, so we cannot avoid functions altogether unless we write the optimization algorithm by hand. We avoid the latter in order not to divert the focus of this text from the data-assimilation problem, and algorithms to solve it, to the problem of how to optimize an objective function.

Again the forward model is that given by Example 2.8, namely (5.1). The `setup` and `truth` sections are similar to the previous programs, except that G , for example, need not be computed here. The auxiliary objective function \mathbf{I} in this case is $\mathbf{l}(\cdot; y)$ from equation (2.21), given by

$$\mathbf{l}(\cdot; y) = \mathbf{J}(\cdot) + \Phi(\cdot; y), \quad (5.9)$$

where

$$\mathbf{J}(u) := \frac{1}{2} \left| C_0^{-\frac{1}{2}} (u_0 - m_0) \right|^2 + \sum_{j=0}^{J-1} \frac{1}{2} \left| \Sigma^{-\frac{1}{2}} (u_{j+1} - \Psi(u_j)) \right|^2 \quad (5.10)$$

and

$$\Phi(u; y) = \sum_{j=0}^{J-1} \frac{1}{2} \left| \Gamma^{-\frac{1}{2}} (y_{j+1} - h(u_{j+1})) \right|^2. \quad (5.11)$$

It is defined in lines 38–45. The auxiliary objective function takes as inputs `(u, y, sigma, gamma, alpha, m0, C0, J)`, and gives output `out = l(u; y)`, where $u \in R^{J+1}$ (given all the other parameters in its definition—the issue of identifying the input to be optimized over is discussed also below).

The initial guess for the optimization algorithm `uu` is taken as a standard normal random vector over \mathbb{R}^{J+1} in line 27. In line 24, a standard normal random matrix of size 100^2 is drawn and thrown away. This is so that one can easily change the input, e.g., to `randn(z)` for $z \in \mathbb{N}$, and induce different random initial vectors `uu` for the optimization algorithm, while keeping the data fixed by the random number seed `sd` set in line 12. The truth `vt` may be used as initial guess by uncommenting line 28. In particular, if the output of the minimization procedure is different for different initial conditions, then it is possible that the objective function $\mathbf{l}(\cdot; y)$ has multiple minima, and hence the posterior distribution $\mathbb{P}(\cdot | y)$ is multimodal. As we have already seen in Figure 3.8, this is certainly true even in the case of scalar deterministic dynamics, when the underlying map gives rise to a chaotic flow.

The MATLAB optimization function `fminsearch` is called in line 32. The *function handle* command `@(u) I(u, ...)` is used to tell `fminsearch` that the objective function \mathbf{I} is to be considered a function of `u`, even though it may take other parameter values as well (in this case, `y, sigma, gamma, alpha, m0, C0`, and `J`). The outputs of `fminsearch` are the value `vmap` such that $\mathbf{I}(vmap)$ is minimum, the value `fval = I(vmap)`, and the `exit flag`, which takes the value 1 if the algorithm has converged. The reader is encouraged to use the `help` command for more details on this and other MATLAB functions used in the notes.

The results of this minimization procedure are plotted in lines 34–35 together with the true value v^\dagger as well as the data y . In Figure 3.9, such results are presented, including two minima that were found with different initial conditions.

```

1 function this=p7
2 clear;set(0,'defaultaxesfontsize',20);format long
3 %% p7.m weak 4DVAR for sin map (Ex. 1.3)
4 %% setup
5
6 J=5;% number of steps
7 alpha=2.5;% dynamics determined by alpha
8 gamma=1e0;% observational noise variance is gamma^2
9 sigma=1;% dynamics noise variance is sigma^2
10 C0=1;% prior initial condition variance
11 m0=0;% prior initial condition mean
12 sd=1;rng(sd);% choose random number seed
13
14 %% truth
15
16 vt(1)=sqrt(C0)*randn;% truth initial condition
17 for j=1:J
18     vt(j+1)=alpha*sin(vt(j))+sigma*randn;% create truth
19     y(j)=vt(j+1)+gamma*randn;% create data
20 end
21
22 %% solution
23
24 randn(100);% try uncommenting or changing the argument for different
25 % initial conditions -- if the result is not the same,
26 % there may be multimodality (e.g. 1 & 100).
27 uu=randn(1,J+1);% initial guess
28 %uu=vt;% truth initial guess option
29
30 % solve with blackbox
31 % exitflag=1 ==> convergence
32 [vmap,fval,exitflag]=fminsearch(@(u)I(u,y,sigma,gamma,alpha,m0,C0,J),uu)
33
34 figure;plot([0:J],vmap,'Linewidth',2);hold;plot([0:J],vt,'r','Linewidth',2)
35 plot([1:J],y,'g','Linewidth',2);hold;xlabel('j');legend('MAP','truth','y')
36
37 %% auxiliary objective function definition
38 function out=I(u,y,sigma,gamma,alpha,m0,C0,J)
39
40 Phi=0;JJ=1/2/C0*(u(1)-m0)^2;
41 for j=1:J
42     JJ=JJ+1/2/sigma^2*(u(j+1)-alpha*sin(u(j)))^2;
43     Phi=Phi+1/2/gamma^2*(y(j)-u(j+1))^2;
44 end
45 out=Phi+JJ;
```

5.3 Chapter 4 Programs

The programs `p8.m-p15.m`, used to generate the figures in Chapter 4, are presented in this section. Various filtering algorithms used to sample the posterior filtering distribution are given, involving both Gaussian approximation and particle approximation. Since these algorithms are run for very large times (large J), they will be divided into only two sections: `setup`, in which the parameters are defined, and `solution`, in which *both* the truth and observations are generated, *and* the online assimilation of the current observation into the filter solution is performed. The generation of truth can be separated into a `truth` section as in the previous sections, but two loops of length J would be required, and loops are inefficient in MATLAB, so the present format is preferred. The programs in this section are all very similar, and their output is also similar, giving rise to Figures 4.3–4.12. With the exception of `p8.m` and `p9.m`, the forward model is given by Example 2.8 (5.1), and the output is identical, given for `p10.m` through `p15.m` in Figures 4.5–4.7 and 4.8–4.10. Figures 4.11 and 4.12 compare the filters from the other Figures. The program `p8.m` features a two-dimensional linear forward model, and `p9.m` features the forward model from Example 2.9 (5.2). At the end of each program, the outputs are used to plot the mean and the covariance as well as the mean-square error of the filter as functions of the iteration number j .

5.3.1. p8.m

The first filtering program is `p8.m`, which contains an implementation of the Kalman filter applied to Example 2.2,

$$v_{j+1} = Av_j + \xi_j, \quad \text{with } A = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix},$$

and observed data given by

$$y_{j+1} = Hv_{j+1} + \eta_{j+1}$$

with $H = (1, 0)$ and Gaussian noise. Thus only the first component of v_j is observed.

The parameters and initial condition are defined in the `setup` section, lines 3–19. The vectors \mathbf{v} , $\mathbf{m} \in \mathbb{R}^{N \times J}$, $\mathbf{y} \in \mathbb{R}^J$, and $\mathbf{c} \in \mathbb{R}^{N \times N \times J}$ are preallocated to hold the truth, mean, observations, and covariance over the J observation times defined in line 5. In particular, notice that the true initial condition is drawn from $N(m_0, C_0)$ in line 16, where $m_0 = 0$ and $C_0 = 1$ are defined in lines 10–11. The initial *estimate* of the distribution is defined in lines 17–18 as $N(m_0, C_0)$, where $m_0 \sim N(0, 100I)$ and $C_0 \leftarrow 100C_0$, so that the code may test the ability of the filter to lock onto the true distribution, asymptotically in j , given a poor initial estimate. That is to say, the values of (m_0, C_0) are *changed* such that the initial condition is *not* drawn from this distribution.

The main `solution` loop then follows in lines 21–34. The truth \mathbf{v} and the data that are being assimilated \mathbf{y} are sequentially generated within the loop, in lines 24–25. The filter prediction step, in lines 27–28, consists in computing the predictive mean and covariance \hat{m}_j and \hat{C}_j as defined in (4.4) and (4.5) respectively:

$$\hat{m}_{j+1} = Am_j, \quad \hat{C}_{j+1} = AC_jA^T + \Sigma.$$

Notice that indices are not used for the transient variables `mhat` and `chat` representing \hat{m}_j and \hat{C}_j , because they will not be saved from one iteration to the next. In lines 30–33, we implement the analysis formulas for the Kalman filter from Corollary 4.2. In particular, the innovation between the observation of the predicted mean and the actual observation, as introduced in Corollary 4.2, is first computed in line 30,

$$d_j = y_j - H\hat{m}_j. \quad (5.12)$$

Again `d`, which represents d_j , does not have any index for the same reason as above. Next, the Kalman gain defined in Corollary 4.2 is computed in line 31:

$$K_j = \hat{C}_j H^T (H\hat{C}_j H^T + F)^{-1}. \quad (5.13)$$

Once again, an index j is not used for the transient variable `K` representing K_j . Notice that a “forward slash” / is used to compute `B/A=B A^-1`. This is an internal function of MATLAB that will analyze the matrices `B` and `A` to determine an “optimal” method for inversion, given their structure. The update given in Corollary 4.2 is completed in lines 30–32 with the equations

$$m_j = \hat{m}_j + K_j d_j \quad \text{and} \quad C_j = (I - K_j H) \hat{C}_j. \quad (5.14)$$

Finally, in lines 36–50, the outputs of the program are used to plot the mean and the covariance as well as the mean-square error of the filter as functions of the iteration number j , as shown in Figure 4.3.

```

1 clear;set(0,'defaultaxesfontsize',20);format long
2 %% p8.m Kalman Filter, Ex. 1.2
3 %% setup
4
5 J=1e3;% number of steps
6 N=2;% dimension of state
7 I=eye(N);% identity operator
8 gamma=1;% observational noise variance is gamma^2*I
9 sigma=1;% dynamics noise variance is sigma^2*I
10 C0=eye(2);% prior initial condition variance
11 m0=[0;0];% prior initial condition mean
12 sd=10;rng(sd);% choose random number seed
13 A=[0 1;-1 0];% dynamics determined by A
14
15 m=zeros(N,J);v=m;y=zeros(J,1);c=zeros(N,N,J);% pre-allocate
16 v(:,1)=m0+sqrtm(C0)*randn(N,1);% initial truth
17 m(:,1)=10*randn(N,1);% initial mean/estimate
18 c(:,:,1)=100*C0;% initial covariance
19 H=[1,0];% observation operator
20
21 %% solution % assimilate!
22
23 for j=1:J
24     v(:,j+1)=A*v(:,j) + sigma*randn(N,1);% truth
25     y(j)=H*v(:,j+1)+gamma*randn;% observation
26
27     mhat=A*m(:,j);% estimator predict
28     chat=A*c(:,:,j)*A'+sigma^2*I;% covariance predict
29
30     d=y(j)-H*mhat;% innovation
31     K=(chat*H')/(H*chat*H'+gamma^2);% Kalman gain
32     m(:,j+1)=mhat+K*d;% estimator update
33     c(:,:,j+1)=(I-K*H)*chat;% covariance update
34 end
35
36 figure;js=21;plot([0:js-1],v(2,1:js));hold;plot([0:js-1],m(2,1:js),'m');
37 plot([0:js-1],m(2,1:js)+reshape(sqrt(c(2,2,1:js)),1,js),'r--');
38 plot([0:js-1],m(2,1:js)-reshape(sqrt(c(2,2,1:js)),1,js),'r--');
39 hold;grid;xlabel('iteration, j');
40 title('Kalman Filter, Ex. 1.2');
41
42 figure;plot([0:J],reshape(c(1,1,:)+c(2,2,:),J+1,1));hold
43 plot([0:J],cumsum(reshape(c(1,1,:)+c(2,2,:),J+1,1))./[1:J+1],'m', ...
44 'Linewidth',2); grid; hold;xlabel('iteration, j');axis([1 1000 0 50]);
45 title('Kalman Filter Covariance, Ex. 1.2');
46
47 figure;plot([0:J],sum((v-m).^2));hold;
48 plot([0:J],cumsum(sum((v-m).^2))./[1:J+1],'m','Linewidth',2);grid
49 hold;xlabel('iteration, j');axis([1 1000 0 50]);
50 title('Kalman Filter Error, Ex. 1.2')

```

5.3.2. p9.m

The program `p9.m` contains an implementation of the 3DVAR method applied to the chaotic logistic map of Example 2.4 (5.2) for $r = 4$. As in the previous section, the parameters and initial condition are defined in the `setup` section, lines 3–16. In particular, notice that the truth initial condition $v(1)$ and initial mean $m(1)$ are now initialized in lines 12–13 with a *uniform* random number using the command `rand`, so that they are in the interval $[0, 1]$, where the model is well defined. Indeed, the solution will eventually become unbounded if initial conditions are chosen outside this interval. With this in mind, we set the dynamics noise `sigma = 0` in line 8, i.e., deterministic dynamics, so that the true dynamics themselves do not become unbounded.

The analysis step of 3DVAR consists in minimizing

$$l_{\text{filter}}(v) = \frac{1}{2}|\Gamma^{-\frac{1}{2}}(y_{j+1} - Hv)|^2 + \frac{1}{2}|\hat{C}^{-\frac{1}{2}}(v - \Psi(m_j))|^2.$$

In this one-dimensional case, we set $\Gamma = \gamma^2$, $\hat{C} = \sigma^2$ and define $\eta^2 = \gamma^2/\sigma^2$. The stabilization parameter η (`eta`) from Example 4.12 is set in line 14, representing the ratio in uncertainty in the data to that of the model; equivalently, it measures trust in the model over the observations. The choice $\eta = 0$ means that the model is irrelevant in the minimization step (4.12) of 3DVAR, in the observed space—the synchronization filter. Since in the example, the signal space and observation space both have dimension equal to 1, the choice $\eta = 0$ simply corresponds to using only the data. In contrast, the choice $\eta = \infty$ ignores the observations and uses only the model.

The 3DVAR setup gives rise to the constant scalar covariance `C` and resultant constant scalar gain `K`; this should not be confused with the changing K_j in (5.13), temporarily defined by `K` in line 31 of `p8.m`. The main `solution` loop follows in lines 20–33. Up to the different forward model, lines 21–22, 24, 26, and 27 of this program are identical to lines 24–25, 27, 30, and 32 of `p8.m`, described in Section 5.3.1. The only other difference is that the covariance updates are not here because of the constant-covariance assumption underlying the 3DVAR algorithm.

The 3DVAR filter may in principle generate the estimated mean `mhat` outside $[0, 1]$, because of the noise in the data. In order to flag potential unbounded trajectories of the filter, which in principle could arise because of this, an extra stopping criterion is included in lines 29–32. To illustrate this, try setting `sigma=0` in line 8. Then the signal will eventually become unbounded, regardless of how small the noise variance is chosen. In this case, the estimate will surely blow up while tracking the unbounded signal. Otherwise, if η is chosen appropriately so as to stabilize the filter, it is extremely unlikely that the estimate will ever blow up. Finally, similarly to `p8.m`, in the last lines of the program we use the outputs of the program in order to produce Figure 4.4, namely plotting the mean and the covariance as well as the mean-square error of the filter as functions of the iteration number j .

```

1 clear;set(0,'defaultaxesfontsize',20);format long
2 %% p9.m 3DVAR Filter, deterministic logistic map (Ex. 1.4)
3 %% setup
4
5 J=1e3;% number of steps
6 r=4;% dynamics determined by r
7 gamma=1e-1;% observational noise variance is gamma^2
8 sigma=0;% dynamics noise variance is sigma^2
9 sd=10;rng(sd);% choose random number seed
10
11 m=zeros(J,1);v=m;y=m;% pre-allocate
12 v(1)=rand;% initial truth, in [0,1]
13 m(1)=rand;% initial mean/estimate, in [0,1]
14 eta=2e-1;% stabilization coefficient 0 < eta << 1
15 C=gamma^2/eta;H=1;% covariance and observation operator
16 K=(C*H')/(H*C*H'+gamma^2);% Kalman gain
17
18 %% solution % assimilate!
19
20 for j=1:J
21     v(j+1)=r*v(j)*(1-v(j)) + sigma*randn;% truth
22     y(j)=H*v(j+1)+gamma*randn;% observation
23
24     mhat=r*m(j)*(1-m(j));% estimator predict
25
26     d=y(j)-H*mhat;% innovation
27     m(j+1)=mhat+K*d;% estimator update
28
29     if norm(mhat)>1e5
30         disp('blowup!')
31         break
32     end
33 end
34 js=21;% plot truth, mean, standard deviation, observations
35 figure;plot([0:js-1],v(1:js));hold;plot([0:js-1],m(1:js),'m');
36 plot([0:js-1],m(1:js)+sqrt(C),'r--');plot([1:js-1],y(1:js-1),'kx');
37 plot([0:js-1],m(1:js)-sqrt(C),'r--');hold;grid;xlabel('iteration, j');
38 title('3DVAR Filter, Ex. 1.4')
39
40 figure;plot([0:J],C*[0:J].^0);hold
41 plot([0:J],C*[0:J].^0,'m','LineWidth',2);grid
42 hold;xlabel('iteration, j');title('3DVAR Filter Covariance, Ex. 1.4');
43
44 figure;plot([0:J],(v-m).^2);hold;
45 plot([0:J],cumsum((v-m).^2)./[1:J+1],'m','LineWidth',2);grid
46 hold;xlabel('iteration, j');
47 title('3DVAR Filter Error, Ex. 1.4')

```

5.3.3. p10.m

A variation of program `p9.m` is given by `p10.m`, where the 3DVAR filter is implemented for Example 2.3 given by (5.1). Indeed, the remaining programs of this section will all be for the same example, namely Example 2.3, so this will not be mentioned again. In this case, the initial condition is again taken as a draw from the prior $N(m_0, C_0)$ as in `p7.m`, and the initial mean estimate is again *changed* to $m_0 \sim N(0, 100I)$ so that the code may test the ability of the filter to lock onto the signal given a poor initial estimate. Furthermore, for this problem, there is no need to introduce the stopping criterion present in the case of `p9.m` since the underlying deterministic dynamics are dissipative. The output of this program is shown in Figure 4.5.

```

1 clear;set(0,'defaultaxesfontsize',20);format long
2 %% p10.m 3DVAR Filter, sin map (Ex. 1.3)
3 % setup
4
5 J=1e3;% number of steps
6 alpha=2.5;% dynamics determined by alpha
7 gamma=1;% observational noise variance is gamma^2
8 sigma=3e-1;% dynamics noise variance is sigma^2
9 C0=9e-2;% prior initial condition variance
10 m0=0;% prior initial condition mean
11 sd=1;rng(sd);% choose random number seed
12
13 m=zeros(J,1);v=m;% pre-allocate
14 v(1)=m0+sqrt(C0)*randn;% initial truth
15 m(1)=10*randn;% initial mean/estimate
16 eta=2e-1;% stabilization coefficient 0 < eta << 1
17 c=gamma^2/eta;H=1;% covariance and observation operator
18 K=(c*H')/(H*c*H'+gamma^2);% Kalman gain
19
20 %% solution % assimilate!
21
22 for j=1:J
23     v(j+1)=alpha*sin(v(j)) + sigma*randn;% truth
24     y(j)=H*v(j+1)+gamma*randn;% observation
25
26     mhat=alpha*sin(m(j));% estimator predict
27
28     d=y(j)-H*mhat;% innovation
29     m(j+1)=mhat+K*d;% estimator update
30
31 end
32
33 js=21;% plot truth, mean, standard deviation, observations
34 figure;plot([0:js-1],v(1:js));hold;plot([0:js-1],m(1:js),'m');
35 plot([0:js-1],m(1:js)+sqrt(c),'r--');plot([1:js-1],y(1:js-1),'kx');
36 plot([0:js-1],m(1:js)-sqrt(c),'r--');hold;grid;xlabel('iteration, j');
37 title('3DVAR Filter, Ex. 1.3')
38
39 figure;plot([0:J],c*[0:J].^0);hold
40 plot([0:J],c*[0:J].^0,'m','LineWidth',2);grid
41 hold;xlabel('iteration, j');
42 title('3DVAR Filter Covariance, Ex. 1.3');
43
44 figure;plot([0:J],(v-m).^2);hold;
45 plot([0:J],cumsum((v-m).^2)./[1:J+1]','m','LineWidth',2);grid
46 hold;xlabel('iteration, j');
47 title('3DVAR Filter Error, Ex. 1.3')

```

5.3.4. p11.m

The next program is `p11.m`. This program comprises an implementation of the extended Kalman filter. It is very similar in structure to `p8.m`, except with a different forward model. Since the dynamics are scalar, the observation operator is defined by setting H to take the value 1 in line 16. The predicting covariance \hat{C}_j is not independent of the mean, as it is for the linear problem `p8.m`. Instead, as described in Section 4.2.2, it is determined via the *linearization* of the forward map around m_j , in line 26:

$$\hat{C}_{j+1} = (\alpha \cos(m_j)) C_j (\alpha \cos(m_j)).$$

As in `p8.m`, we *change* the prior to a poor initial estimate of the distribution to study whether, and how, the filter locks onto a neighborhood of the true signal, despite poor initialization, for large j . This initialization is in lines 15–16, where $m_0 \sim N(0, 100I)$ and $C_0 \leftarrow 10C_0$. Subsequent filtering programs use an identical initialization, with the same rationale as in this case. We will not state this again. The output of this program is shown in Figure 4.6.

```

1 clear;set(0,'defaultaxesfontsize',20);format long
2 %% p11.m Extended Kalman Filter, sin map (Ex. 1.3)
3 %% setup
4
5 J=1e3;% number of steps
6 alpha=2.5;% dynamics determined by alpha
7 gamma=1;% observational noise variance is gamma^2
8 sigma=3e-1;% dynamics noise variance is sigma^2
9 C0=9e-2;% prior initial condition variance
10 m0=0;% prior initial condition mean
11 sd=1;rng(sd);% choose random number seed
12
13 m=zeros(J,1);v=m;c=m;% pre-allocate
14 v(1)=m0+sqrt(C0)*randn;% initial truth
15 m(1)=10*randn;% initial mean/estimate
16 c(1)=10*C0;H=1;% initial covariance and observation operator
17
18 %% solution % assimilate!
19
20 for j=1:J
21
22     v(j+1)=alpha*sin(v(j)) + sigma*randn;% truth
23     y(j)=H*v(j+1)+gamma*randn;% observation
24
25     mhat=alpha*sin(m(j));% estimator predict
26     chat=alpha*cos(m(j))*c(j)*alpha*cos(m(j))+sigma^2;% covariance predict
27
28     d=y(j)-H*mhat;% innovation
29     K=(chat*H')/(H*chat*H'+gamma^2);% Kalman gain
30     m(j+1)=mhat+K*d;% estimator update
31     c(j+1)=(1-K*H)*chat;% covariance update
32
33 end
34
35 js=21;% plot truth, mean, standard deviation, observations
36 figure;plot([0:js-1],v(1:js));hold;plot([0:js-1],m(1:js),'m');
37 plot([0:js-1],m(1:js)+sqrt(c(1:js)),'r--');plot([1:js-1],y(1:js-1),'kx');
38 plot([0:js-1],m(1:js)-sqrt(c(1:js)),'r--');hold;grid;xlabel
39 ('iteration, j');
40 title('ExKF, Ex. 1.3')
41
42 figure;plot([0:J],c);hold
43 plot([0:J],cumsum(c)./[1:J+1],'m','Linewidth',2);grid
44 hold;xlabel('iteration, j');
45 title('ExKF Covariance, Ex. 1.3');
46
47 figure;plot([0:J],(v-m).^2);hold;
48 plot([0:J],cumsum((v-m).^2)./[1:J+1],'m','Linewidth',2);grid
49 hold;xlabel('iteration, j');
50 title('ExKF Error, Ex. 1.3')

```

5.3.5. p12.m

The program `p12.m` contains an implementation of the ensemble Kalman filter, with perturbed observations, as described in Section 4.2.3. The structure of this program is again very similar to those of `p8.m` and `p11.m`, except now an ensemble of particles, of size N defined in line 12, is retained as an approximation of the filtering distribution. The ensemble $\{v^{(n)}\}_{n=1}^N$ represented by the matrix U is then constructed out of draws from this Gaussian in line 18, and the mean m'_0 is reset to the ensemble sample mean.

In line 27, the predicting ensemble $\{\hat{v}_j^{(n)}\}_{n=1}^N$ represented by the matrix U_{hat} is computed from a realization of the forward map applied to each ensemble member. This is then used to compute the ensemble sample mean \hat{m}_j (`mhat`) and covariance \hat{C}_j (`chat`). There is now an ensemble of “innovations” with a new i.i.d. realization $y_j^{(n)} \sim N(y_j, \Gamma)$ for each ensemble member, computed in line 31 (not to be confused with the actual innovation as defined in equation (5.12)),

$$d_j^{(n)} = y_j^{(n)} - H\hat{v}_j^{(n)}.$$

The Kalman gain K_j (`K`) is computed using (5.13), very similarly to how it is done in `p8.m` and `p11.m`, and the ensemble of updates is computed in line 33:

$$v_j^{(n)} = \hat{v}_j^{(n)} + K_j d_j^{(n)}.$$

The output of this program is shown in Figure 4.7. Furthermore, long simulations of length $J = 10^5$ were performed for this and the previous two programs, `p10.m` and `p11.m`, and their errors are compared in Figure 4.11.

```

1 clear;set(0,'defaultaxesfontsize',20);format long
2 %% p12.m Ensemble Kalman Filter (PO), sin map (Ex. 1.3)
3 %% setup
4
5 J=1e5;% number of steps
6 alpha=2.5;% dynamics determined by alpha
7 gamma=1;% observational noise variance is gamma^2
8 sigma=3e-1;% dynamics noise variance is sigma^2
9 C0=9e-2;% prior initial condition variance
10 m0=0;% prior initial condition mean
11 sd=1;rng(sd);% choose random number seed
12 N=10;% number of ensemble members
13
14 m=zeros(J,1);v=m;y=m;c=m;U=zeros(J,N);% pre-allocate
15 v(1)=m0+sqrt(C0)*randn;% initial truth
16 m(1)=10*randn;% initial mean/estimate
17 c(1)=10*C0;H=1;% initial covariance and observation operator
18 U(1,:)=m(1)+sqrt(c(1))*randn(1,N);m(1)=sum(U(1,:))/N;% initial ensemble
19
20 %% solution % assimilate!
21
22 for j=1:J
23
24     v(j+1)=alpha*sin(v(j)) + sigma*randn;% truth
25     y(j)=H*v(j+1)+gamma*randn;% observation
26
27     Uhat=alpha*sin(U(j,:))+sigma*randn(1,N);% ensemble predict
28     mhat=sum(Uhat)/N;% estimator predict
29     chat=(Uhat-mhat)*(Uhat-mhat) '/(N-1);% covariance predict
30
31     d=y(j)+gamma*randn(1,N)-H*Uhat;% innovation
32     K=(chat*H')/(H*chat*H'+gamma^2);% Kalman gain
33     U(j+1,:)=Uhat+K*d;% ensemble update
34     m(j+1)=sum(U(j+1,:))/N;% estimator update
35     c(j+1)=(U(j+1,:)-m(j+1))*(U(j+1,:)-m(j+1))'/(N-1);% covariance update
36
37 end
38
39 js=21;% plot truth, mean, standard deviation, observations
40 figure;plot([0:js-1],v(1:js));hold;plot([0:js-1],m(1:js),'m');
41 plot([0:js-1],m(1:js)+sqrt(c(1:js)), 'r--');plot([1:js-1],y(1:js-1), 'kx');
42 plot([0:js-1],m(1:js)-sqrt(c(1:js)), 'r--');hold;grid;xlabel
43 ('iteration, j');
44 title('EnKF, Ex. 1.3')
45
46 figure;plot([0:J],c);hold
47 plot([0:J],cumsum(c)./[1:J+1]', 'm', 'Linewidth', 2);grid
48 hold;xlabel('iteration, j');
49 title('EnKF Covariance, Ex. 1.3');
50
51 figure;plot([0:J],(v-m).^2);hold;
52 plot([0:J],cumsum((v-m).^2)./[1:J+1]', 'm', 'Linewidth', 2);grid
53 hold;xlabel('iteration, j');
54 title('EnKF Error, Ex. 1.3')

```

5.3.6. p13.m

The program `p13.m` contains a particular square-root filter implementation of the ensemble Kalman filter, namely the ETKF filter, described in detail in Section 4.2.4. The program thus is very similar to `p12.m` for the EnKF with perturbed observations. In particular, the filtering distribution of the state is again approximated by an ensemble of particles. The predicting ensemble $\{\hat{v}_j^{(n)}\}_{n=1}^N$ (`Uhat`), mean \hat{m}_j (`mhat`), and covariance \hat{C}_j (`chat`) are computed exactly as in `p12.m`. However, this time the covariance is kept in factorized form $\hat{X}_j \hat{X}_j^\top = \hat{C}_j$ in lines 29–30, with factors denoted by `Xhat`. The transformation matrix is computed in line 31,

$$T_j = \left(I_N + \hat{X}_j^\top H^\top \Gamma^{-1} H \hat{X}_j \right)^{-\frac{1}{2}},$$

and $X_j = \hat{X}_j T_j$ (`X`) is computed in line 32, from which the covariance $C_j = X_j X_j^\top$ is reconstructed in line 38. A single innovation d_j is computed in line 34, and a single updated mean m_j is then computed in line 36 using the Kalman gain K_j (5.13) computed in line 35. This is the same as in the Kalman filter and extended Kalman filter (ExKF) of `p8.m` and `p11.m`, in contrast to the EnKF with perturbed observations appearing in `p12.m`. The ensemble is then updated to `U` in line 37 using the formula

$$v_j^{(n)} = m_j + X_j^{(n)} \sqrt{N-1},$$

where $X_j^{(n)}$ is the n th column of X_j .

Notice that the operator that is factorized and inverted has dimension N , which in this case is large in comparison to the state and observation dimensions. This is, of course, natural for computing sample statistics, but in the context of the one-dimensional examples considered here, it makes `p13.m` run far more slowly than `p12.m`. However, in many applications, the signal state-space dimension is the largest, with the observation dimension coming next, and the ensemble size being far smaller than either of these. In this context, the ETKF has become a very popular method. So its relative inefficiency, compared, for example, with the perturbed observations Kalman filter, should not be given too much weight in the overall evaluation of the method. Results illustrating the algorithm are shown in Figure 4.8.

```

1 clear;set(0,'defaultaxesfontsize',20);format long
2 %% p13.m Ensemble Kalman Filter (ETKF), sin map (Ex. 1.3)
3 %% setup
4
5 J=1e3;% number of steps
6 alpha=2.5;% dynamics determined by alpha
7 gamma=1;% observational noise variance is gamma^2
8 sigma=3e-1;% dynamics noise variance is sigma^2
9 C0=9e-2;% prior initial condition variance
10 m0=0;% prior initial condition mean
11 sd=1;rng(sd);% choose random number seed
12 N=10;% number of ensemble members
13
14 m=zeros(J,1);v=m;y=m;c=m;U=zeros(J,N);% pre-allocate
15 v(1)=m0+sqrt(C0)*randn;% initial truth
16 m(1)=10*randn;% initial mean/estimate
17 c(1)=10*C0;H=1;% initial covariance and observation operator
18 U(1,:)=m(1)+sqrt(c(1))*randn(1,N);m(1)=sum(U(1,:))/N;% initial ensemble
19
20 %% solution % assimilate!
21
22 for j=1:J
23
24     v(j+1)=alpha*sin(v(j)) + sigma*randn;% truth
25     y(j)=H*v(j+1)+gamma*randn;% observation
26
27     Uhat=alpha*sin(U(j,:))+sigma*randn(1,N);% ensemble predict
28     mhat=sum(Uhat)/N;% estimator predict
29     Xhat=(Uhat-mhat)/sqrt(N-1);% centered ensemble
30     chat=Xhat*Xhat';% covariance predict
31     T=sqrtn(inv(eye(N)+Xhat'*H'*H*Xhat/gamma^2));% right-hand sqrt
32         transform
33     X=Xhat*T;% transformed centered ensemble
34
35     d=y(j)-H*mhat;randn(1,N);% innovation
36     K=(chat*H')/(H*chat*H'+gamma^2);% Kalman gain
37     m(j+1)=mhat+K*d;% estimator update
38     U(j+1,:)=m(j+1)+X*sqrt(N-1);% ensemble update
39     c(j+1)=X*X';% covariance update
40
41 end
42
43 js=21;% plot truth, mean, standard deviation, observations
44 figure;plot([0:js-1],v(1:js));hold;plot([0:js-1],m(1:js),'m');
45 plot([0:js-1],m(1:js)+sqrt(c(1:js)),'r--');plot([1:js-1],y(1:js-1),'kx');
46 plot([0:js-1],m(1:js)-sqrt(c(1:js)),'r--');hold;grid;xlabel
47 ('iteration, j');
48 title('EnKF(ETKF), Ex. 1.3');
49
50 figure;plot([0:J],(v-m).^2);hold;
51 plot([0:J],cumsum((v-m).^2)./[1:J+1]', 'm', 'Linewidth', 2);grid
52 plot([0:J],cumsum(c)./[1:J+1]', 'r--', 'Linewidth', 2);
53 hold;xlabel('iteration, j');
54 title('EnKF(ETKF) Error, Ex. 1.3')

```

5.3.7. p14.m

The program `p14.m` is an implementation of the standard SIRS filter from Section 4.3.2. The `setup` section is almost identical to those of the EnKF methods, because those methods also rely on particle approximations of the filtering distribution. However, the particle filters consistently estimate quite general distributions, while the EnKF is provably accurate only for Gaussian distributions. The truth and data generation and ensemble prediction in lines 24–27 are the same as in `p12.m` and `p13.m`. The way this prediction in line 27 is phrased in Section 4.3.2 is $\hat{v}_{j+1}^{(n)} \sim \mathbb{P}(\cdot | v_j^{(n)})$. An ensemble of “innovation” terms $\{d_j^{(n)}\}_{n=1}^N$ is again required, but with all terms using the *same* observation, as computed in line 28. Assuming $w_j^{(n)} = 1/N$, then

$$\hat{w}_j^{(n)} \propto \mathbb{P}(y_j | v_j^{(n)}) \propto \exp \left\{ -\frac{1}{2} \left| d_j^{(n)} \right|_\Gamma^2 \right\},$$

where $d_j^{(n)}$ is the innovation of the n th particle, as given in (4.27). The vector of unnormalized weights $\{\hat{w}_j^{(n)}\}_{n=1}^N$ (`what`) is computed in line 29 and normalized to $\{w_j^{(n)}\}_{n=1}^N$ (`w`) in line 30. Lines 32–39 implement the resampling step. First, the cumulative distribution function of the weights $W \in [0, 1]^N$ (`ws`) is computed in line 32. Notice that W has the properties $W_1 = w_j^{(1)}$, $W_n \leq W_{n+1}$, and $W_N = 1$. Then N uniform random numbers $\{u^{(n)}\}_{n=1}^N$ are drawn. For each $u^{(n)}$, let n^* be such that $W_{n^*-1} \leq u^{(n)} < W_{n^*}$. This n^* (`ix`) is found in line 34 using the `find` function, which can identify the first or last element in an array to exceed zero (see help file): `ix = find (ws > rand, 1, 'first')`. This corresponds to drawing the (n^*) th element from the discrete measure defined by $\{w_j^{(n)}\}_{n=1}^N$. The n th particle $v_j^{(n)}$ (`U(j+1, n)`) is set equal to $\hat{v}_j^{(n^*)}$ (`Uhat(ix)`) in line 37. The sample mean and covariance are then computed in lines 41–42. The rest of the program follows the others, generating the output displayed in Figure 4.9.

```

1 clear;set(0,'defaultaxesfontsize',20);format long
2 %% p14.m Particle Filter (SIRS), sin map (Ex. 1.3)
3 %% setup
4
5 J=1e3;% number of steps
6 alpha=2.5;% dynamics determined by alpha
7 gamma=1;% observational noise variance is gamma^2
8 sigma=3e-1;% dynamics noise variance is sigma^2
9 C0=9e-2;% prior initial condition variance
10 m0=0;% prior initial condition mean
11 sd=1;rng(sd);% choose random number seed
12 N=100;% number of ensemble members
13
14 m=zeros(J,1);v=m;y=m;c=m;U=zeros(J,N);% pre-allocate
15 v(1)=m0+sqrt(C0)*randn;% initial truth
16 m(1)=10*randn;% initial mean/estimate
17 c(1)=10*C0;H=1;% initial covariance and observation operator
18 U(1,:)=m(1)+sqrt(c(1))*randn(1,N);m(1)=sum(U(1,:))/N;% initial ensemble
19
20 %% solution % Assimilate!
21
22 for j=1:J
23
24     v(j+1)=alpha*sin(v(j)) + sigma*randn;% truth
25     y(j)=H*v(j+1)+gamma*randn;% observation
26
27     Uhat=alpha*sin(U(j,:))+sigma*randn(1,N);% ensemble predict
28     d=y(j)-H*Uhat;% ensemble innovation
29     what=exp(-1/2*(1/gamma^2*d.^2));% weight update
30     w=what/sum(what);% normalize predict weights
31
32     ws=cumsum(w);% resample: compute cdf of weights
33     for n=1:N
34         ix=find(ws>rand,1,'first');% resample: draw rand \sim U[0,1] and
35         % find the index of the particle corresponding to the first time
36         % the cdf of the weights exceeds rand.
37         U(j+1,n)=Uhat(ix);% resample: reset the nth particle to the one
38         % with the given index above
39     end
40
41     m(j+1)=sum(U(j+1,:))/N;% estimator update
42     c(j+1)=(U(j+1,:)-m(j+1))*(U(j+1,:)-m(j+1))'/N;% covariance update
43
44 end
45
46 js=21;% plot truth, mean, standard deviation, observations
47 figure;plot([0:js-1],v(1:js));hold;plot([0:js-1],m(1:js),'m');
48 plot([0:js-1],m(1:js)+sqrt(c(1:js)),'r--');plot([1:js-1],y(1:js-1),'kx');
49 plot([0:js-1],m(1:js)-sqrt(c(1:js)),'r--');hold;grid;xlabel
50 ('iteration, j');
51 title('Particle Filter (Standard), Ex. 1.3');
52
53 figure;plot([0:J],(v-m).^2);hold;
54 plot([0:J],cumsum((v-m).^2)./[1:J+1]','m','Linewidth',2);grid
55 hold;xlabel('iteration, j');title('Particle Filter (Standard) Error,
56 Ex. 1.3')

```

5.3.8. p15.m

The program `p15.m` is an implementation of the SIRS(OP) algorithm from Section 4.3.3. The setup section and truth and observation generation are again the same as in the previous programs. The difference between this program and `p14.m` arises because the importance sampling proposal kernel Q_j with density $\mathbb{P}(v_{j+1}|v_j, y_{j+1})$ is used to propose each $\hat{v}_{j+1}^{(n)}$ given each particular $v_j^{(n)}$; in particular, Q_j depends on the next data point, whereas the kernel P used in `p14.m` has density $\mathbb{P}(v_{j+1}|v_j)$, which is independent of y_{j+1} .

Observe that if $v_j^{(n)}$ and y_{j+1} are both fixed, then $\mathbb{P}\left(v_{j+1}|v_j^{(n)}, y_{j+1}\right)$ is the density of the Gaussian with mean $m'^{(v)}$ and covariance Σ' given by

$$m'^{(n)} = \Sigma' \left(\Sigma^{-1} \Psi\left(v_j^{(n)}\right) + H^\top \Gamma^{-1} y_{j+1} \right), \quad (\Sigma')^{-1} = \Sigma^{-1} + H^\top \Gamma^{-1} H.$$

Therefore, Σ' (`Sig`) and the ensemble of means $\{m'^{(n)}\}_{n=1}^N$ (vector `em`) are computed in lines 27 and 28 and used to sample $\hat{v}_{j+1}^{(n)} \sim N(m'^{(n)}, \Sigma')$ in line 29 for all of $\{\hat{v}_{j+1}^{(n)}\}_{n=1}^N$ (`Uhat`).

Now the weights are updated by (4.34) rather than (4.27), i.e., assuming $w_j^{(n)} = 1/N$. Then

$$\hat{w}_{j+1}^{(n)} \propto \mathbb{P}\left(y_{j+1}|v_j^{(n)}\right) \propto \exp \left\{ -\frac{1}{2} \left| y_{j+1} - \Psi\left(v_j^{(n)}\right) \right|_{\Gamma+\Sigma}^2 \right\}.$$

This is computed in lines 31–32, using another auxiliary “innovation” vector `d` in line 31. Lines 35–45 are again identical to lines 32–42 of program `p14.m`, performing the resampling step and computing the sample mean and covariance.

The output of this program was used to produce Figure 4.10, similar to the other filtering algorithms. Furthermore, long simulations of length $J = 10^5$ were performed for this and the previous three programs, `p12.m`, `p13.m`, and `p14.m`, and their errors are compared in Figure 4.12, similarly to Figure 4.11, comparing the basic filters `p10.m`, `p11.m`, and `p12.m`.

```

1 clear;set(0,'defaultaxesfontsize',20);format long
2 %% p15.m Particle Filter (SIRS, OP), sin map (Ex. 1.3)
3 %% setup
4
5 J=1e3;% number of steps
6 alpha=2.5;% dynamics determined by alpha
7 gamma=1;% observational noise variance is gamma^2
8 sigma=3e-1;% dynamics noise variance is sigma^2
9 C0=9e-2;% prior initial condition variance
10 m0=0;% prior initial condition mean
11 sd=1;rng(sd);% choose random number seed
12 N=100;% number of ensemble members
13
14 m=zeros(J,1);v=m;y=m;c=m;U=zeros(J,N);% pre-allocate
15 v(1)=m0+sqrt(C0)*randn;% initial truth
16 m(1)=10*randn;% initial mean/estimate
17 c(1)=10*C0;H=1;% initial covariance and observation operator
18 U(1,:)=m(1)+sqrt(c(1))*randn(1,N);m(1)=sum(U(1,:))/N;% initial ensemble
19
20 %% solution % Assimilate!
21
22 for j=1:J
23
24     v(j+1)=alpha*sin(v(j)) + sigma*randn;% truth
25     y(j)=H*v(j+1)+gamma*randn;% observation
26
27     Sig=inv(inv(sigma^2)+H'*inv(gamma^2)*H);% optimal proposal covariance
28     em=Sig*(inv(sigma^2)*alpha*sin(U(j,:))+H'*inv(gamma^2)*y(j));% proposal
29         mean
30     Uhat=em+sqrt(Sig)*randn(1,N);% ensemble optimally importance sampled
31
32     d=y(j)-H*alpha*sin(U(j,:));% ensemble innovation
33     what=exp(-1/2/(sigma^2+gamma^2)*d.^2);% weight update
34     w=what/sum(what);% normalize predict weights
35
36     ws=cumsum(w);% resample: compute cdf of weights
37     for n=1:N
38         ix=find(ws>rand,1,'first');% resample: draw rand \sim U[0,1] and
39             % find the index of the particle corresponding to the first time
40             % the cdf of the weights exceeds rand.
41         U(j+1,n)=Uhat(ix);% resample: reset the nth particle to the one
42             % with the given index above
43     end
44
45     m(j+1)=sum(U(j+1,:))/N;% estimator update
46     c(j+1)=(U(j+1,:)-m(j+1))*(U(j+1,:)-m(j+1))'/N;% covariance update
47
48 end
49
50 js=21;%plot truth, mean, standard deviation, observations
51 figure;plot([0:js-1],v(1:js));hold;plot([0:js-1],m(1:js),'m');
52 plot([0:js-1],m(1:js)+sqrt(c(1:js)),r--');plot([1:js-1],y(1:js-1),kx');
53 plot([0:js-1],m(1:js)-sqrt(c(1:js)),r--');hold;grid;xlabel
54 ('iteration, j');
55 title('Particle Filter (Optimal), Ex. 1.3');

```

5.4 ODE Programs

The programs `p16.m` and `p17.m` are used to simulate and plot the Lorenz '63 and '96 models from Examples 2.6 and 2.7, respectively. These programs are both MATLAB functions, similar to the program `p7.m` presented in Section 5.2.5. The reason for using functions and not scripts is that the black box MATLAB built-in function `ode45` can be used for the time integration (see `help` page for details regarding this function). Therefore, each has an *auxiliary* function defining the right-hand side of the given ODE, which is passed via a *function handle* to `ode45`.

5.4.1. p16.m

The first of the ODE programs, `p16.m`, integrates the Lorenz '63 model 2.6. The setup section of the program, on lines 4–11, defines the parameters of the model and the initial conditions. In particular, a random Gaussian initial condition is chosen in line 9, and a small perturbation to its first (x) component is introduced in line 10. The trajectories are computed on lines 13–14 using the built-in function `ode45`. Notice that the auxiliary function `lorenz63`, defined on line 29, takes as arguments (t, y) , prescribed through the definition of the function handle `@(t, y)`, while (α, b, r) are given as fixed parameters `(a, b, r)`, defining the particular instance of the function. The argument t is intended for defining *nonautonomous* ODEs and is spurious here, since it is an *autonomous* ODE, and therefore, t does not appear on the right-hand side. It is nonetheless included for completeness, and it causes no harm. The Euclidean norm of the error is computed in line 16, and the results are plotted similarly to previous programs in lines 18–25. This program is used to plot Figs. 2.6 and 2.7.

5.4.2. p17.m

The second of the ODE programs, `p17.m`, integrates the $J=40$ -dimensional Lorenz '96 model 2.7. This program is almost identical to the previous one, where a small perturbation of the random Gaussian initial condition defined on line 9 is introduced on lines 10–11. The major difference is the function passed to `ode45` on lines 14–15, which now defines the right-hand side of the Lorenz '96 model given by the subfunction `lorenz96` on line 30. Again the system is autonomous, and the spurious t -variable is included for completeness. A few of the 40 degrees of freedom are plotted along with the error in lines 19–27. This program is used to plot Figs. 2.8 and 2.9

```
1 function this=p16
2 clear;set(0,'defaultaxesfontsize',20);format long
3 %% p16.m Lorenz '63 (Ex. 2.6)
4 %% setup
5
6 a=10;b=8/3;r=28;% define parameters
7 sd=1;rng(sd);% choose random number seed
8
9 initial=randn(3,1);% choose initial condition
10 initial1=initial + [0.0001;0;0];% choose perturbed initial condition
11
12 %% calculate the trajectories with blackbox
13 [t1,y]=ode45(@(t,y) lorenz63(t,y,a,b,r), [0 100], initial);
14 [t,y1]=ode45(@(t,y) lorenz63(t,y,a,b,r), t1, initial1);
15
16 error=sqrt(sum((y-y1).^2,2));% calculate error
17
18 %% plot results
19
20 figure(1), semilogy(t,error,'k')
21 axis([0 100 10^-6 10^2])
22 set(gca,'YTick',[10^-6 10^-4 10^-2 10^0 10^2])
23
24 figure(2), plot(t,y(:,1),'k')
25 axis([0 100 -20 20])
26
27
28 %% auxiliary dynamics function definition
29 function rhs=lorenz63(t,y,a,b,r)
30
31 rhs(1,1)=a*(y(2)-y(1));
32 rhs(2,1)=-a*y(1)-y(2)-y(1)*y(3);
33 rhs(3,1)=y(1)*y(2)-b*y(3)-b*(r+a);
```

```
1 function this=p17
2 clear;set(0,'defaultaxesfontsize',20);format long
3 %% p17.m Lorenz '96 (Ex. 2.7)
4 %% setup
5
6 J=40;F=8;% define parameters
7 sd=1;rng(sd);% choose random number seed
8
9 initial=randn(J,1);% choose initial condition
10 initial1=initial;
11 initial1(1)=initial(1)+0.0001;% choose perturbed initial condition
12
13 %% calculate the trajectories with blackbox
14 [t1,y]=ode45(@(t,y) lorenz96(t,y,F), [0 100], initial);
15 [t,y1]=ode45(@(t,y) lorenz96(t,y,F), t1, initial1);
16
17 error=sqrt(sum((y-y1).^2,2));% calculate error
18
19 %% plot results
20
21 figure(1), plot(t,y(:,1),'k')
22 figure(2), plot(y(:,1),y(:,J),'k')
23 figure(3), plot(y(:,1),y(:,J-1),'k')
24
25 figure(4), semilogy(t,error,'k')
26 axis([0 100 10^-6 10^2])
27 set(gca,'YTick',[10^-6 10^-4 10^-2 10^0 10^2])
28
29 %% auxiliary dynamics function definition
30 function rhs=lorenz96(t,y,F)
31
32 rhs=[y(end);y(1:end-1)].*( [y(2:end);y(1)] - ...
33 [y(end-1:end);y(1:end-2)]) - y + F*y.^0;
```

Chapter 6

Continuous Time: Formulation

In this chapter, and in all subsequent chapters, we consider continuous-time signal dynamics and continuous-time data. This takes us into a part of the subject that is potentially rather technical, a fact that can obscure the structure manifest in the continuous-time formulation. In order to avoid technicalities that can obfuscate the derivations, and in order to create space to highlight the structure present in the continuous-time models, we proceed as follows: we adopt an approach whereby the derivation of many key equations proceeds in a nonrigorous fashion from the discrete-time setup, by formally taking the limit $\tau \rightarrow 0$, where τ is the time increment between observations. We then concentrate on studying the properties of the resulting limiting continuous-time problems, and algorithms for them.

This chapter commences with a derivation of the continuous-time setting, via the limiting process $\tau \rightarrow 0$, in Section 6.1; we also include a brief overview of the properties of stochastic integrals and stochastic differential equations (SDEs). Section 6.2 contains the guiding examples that we use throughout the continuous-time part of these notes, and is followed, in Sections 6.3 and 6.4, with a discussion of the filtering and smoothing problems, respectively. As in the discrete-time setting, filtering and smoothing are related, and this fact is discussed, and references given, in the bibliography, Section 6.6. Well-posedness of the distributions is considerably harder to study in continuous time than in discrete time, and we also provide references to the relevant literature on this topic within the bibliography, Section 6.6, rather than analyze it explicitly. Section 6.5 contains numerical illustrations centered on the guiding examples. The chapter concludes with exercises in Section 6.7.

6.1 Setup

6.1.1. Derivation of the Continuous-Time Model

In order to derive a continuous-time limit that exhibits clearly the tension between deterministic and stochastic elements, we consider the discrete-time model (2.1) and (2.2) in the case that for $\tau > 0$,

$$\Psi(\cdot) \rightarrow I \cdot + \tau f(\cdot), \quad y_{j+1} \rightarrow \left(\frac{z_{j+1} - z_j}{\tau} \right), \quad \Sigma \rightarrow \tau \Sigma_0, \quad \text{and} \quad \Gamma \rightarrow \tau^{-1} \Gamma_0. \quad (6.1)$$

Electronic supplementary material The online version of this chapter (doi: [10.1007/978-3-319-20325-6_6](https://doi.org/10.1007/978-3-319-20325-6_6)) contains supplementary material, which is available to authorized users.

The assumption on Ψ , namely that it is a small $\mathcal{O}(\tau)$ perturbation of the identity, and the scaling of the model noise Σ to have variance $\mathcal{O}(\tau)$, ensures that the underlying signal process behaves like an SDE for $\tau \rightarrow 0$. In contrast to the model noise, however, the observational noise variance Γ is scaled *inversely* proportional to τ . This large observational noise scaling counterbalances the effect of increased frequency of observation so as to obtain an interesting effect in the limit. To understand this fact, notice that the variable z is, in the limit, an integral of the observations y . Together, this scaling of the noise and definition of z leads to a situation in which observational noise and the variable z are of the same order of magnitude, and z is governed by an SDE.

From (2.1), the scalings (6.1) give

$$v_{j+1} = v_j + \tau f(v_j) + \sqrt{\tau \Sigma_0} \tilde{\xi}_j, \quad j \in \mathbb{Z}^+, \quad (6.2a)$$

$$v_0 \sim N(m_0, C_0), \quad (6.2b)$$

with $\tilde{\xi} = \{\tilde{\xi}_j\}_{j \in \mathbb{N}}$ an i.i.d. sequence determined by $\tilde{\xi}_0 \sim N(0, I)$ and independent of v_0 . From (2.2), we obtain

$$z_{j+1} = z_j + \tau h(v_{j+1}) + \sqrt{\tau \Gamma_0} \tilde{\eta}_{j+1}, \quad j \in \mathbb{Z}^+, \quad (6.3a)$$

$$z_0 = 0, \quad (6.3b)$$

with $\tilde{\eta} = \{\tilde{\eta}_j\}_{j \in \mathbb{Z}^+}$ an i.i.d. sequence determined by $\tilde{\eta}_1 \sim N(0, I)$. The independence assumptions made in the discrete-time setting imply that $\tilde{\xi}, \tilde{\eta}$ and v_0 are mutually independent random variables. We assume that the pair (v_j, z_j) represent approximations of a continuous-time process $(v(\cdot), z(\cdot))$ evaluated at time $t = j\tau$. Specifically, if we view v_j as an approximation to $v(j\tau)$ and z_j as an approximation to $z(j\tau)$, then (6.2), (6.3) constitute a consistent numerical discretization, with time step τ , of the SDEs

$$\frac{dv}{dt} = f(v) + \sqrt{\Sigma_0} \frac{dB_v}{dt}, \quad (6.4a)$$

$$v(0) \sim N(m_0, C_0), \quad (6.4b)$$

and

$$\frac{dz}{dt} = h(v) + \sqrt{\Gamma_0} \frac{dB_z}{dt}, \quad (6.5a)$$

$$z(0) = 0; \quad (6.5b)$$

here B_v (respectively B_z) is an \mathbb{R}^n -valued (respectively \mathbb{R}^m -valued) standard Brownian motion. In addition, B_v and B_z are independent of each other and of $v(0)$. Note that in fact, (6.2) is the Euler–Maruyama approximation of (6.4), while (6.3) is a semi-implicit approximation of (6.5), of Euler–Maruyama type. Straightforward numerical analysis proves convergence of the solution of (6.2), (6.3) to (6.4), (6.5) under, for example, the assumption that f and h are globally Lipschitz functions. Thus our signal dynamics is determined by (6.4), and the data by (6.5). We will always assume that $\Gamma_0 > 0$. In studying the stochastic dynamics model, we also assume that $\Sigma_0 > 0$. However, we will also consider the deterministic dynamics model for which $\Sigma_0 = 0$.

We now return to the question of why we chose the particular scalings $\Sigma \rightarrow \tau \Sigma_0$ and $\Gamma \rightarrow \tau^{-1} \Gamma_0$. These are the scalings that result in interesting continuous-time limits in which deterministic and stochastic effects are in balance. Choosing $\Sigma \rightarrow \tau^r \Sigma_0$ with $r > 1$ would

result in the deterministic dynamics model considered below, while choosing $r < 1$ would necessitate a different rescaling in time to obtain a continuous limit, and this would simply be Brownian motion. Furthermore, choosing the observational noise variance inversely proportional to the time increment between observations reflects the right balance to see an $\mathcal{O}(1)$ noise polluting the observations in the continuous-time limit; choosing $\Gamma \rightarrow \tau^{-r} \Gamma_0$ with $r < 1$ results in a limiting observation equation like (6.5) but with Γ_0 replaced by 0, and choosing $r > 1$ does not result in an interesting limit, because noise swamps the observations.

This and subsequent chapters will be devoted to the **data-assimilation** problem of finding out information about the signal v solving (6.4) from the data z given by (6.5). Thus our scalings have been chosen so that the resulting continuous-time limit retains the interesting balance between observations and noise that makes the data-assimilation problem challenging; more precisely, it leads to problems in which the fields of stochastic dynamical systems and statistics need to work in conjunction in order to make progress. Our analysis in this and subsequent chapters should mainly be understood as an attempt to understand data assimilation by means of the tools of continuous-time analysis; for some researchers, this brings a clarity to the subject that reveals the key issues more clearly than in discrete time. In particular, this understanding is most pertinent to the understanding of discrete-time data assimilation in the case of high-frequency observations with large observational noise. In rough terms, the analysis is relevant when the high frequency and large noise balance to produce $\mathcal{O}(1)$ stochastic effects.

As in the discrete-time setting, we will be interested, on occasion, in the case of **deterministic signal dynamics**, where $\Sigma_0 \equiv 0$, but we will always have $\Gamma_0 \neq 0$. In the case of deterministic dynamics, we replace equation (6.4) by

$$\frac{dv}{dt} = f(v), \quad (6.6a)$$

$$v(0) \sim N(m_0, C_0). \quad (6.6b)$$

6.1.2. Stochastic Integration

We conclude this section with a brief summary of the basic properties of Itô stochastic integrals and SDEs, since these are used throughout our continuous-time developments. In order to define SDEs, we need to be able to make sense of the stochastic integral

$$\text{SI} := \int_a^b r(t)dW(t), \quad (6.7)$$

where W is a Brownian motion. We now make this idea more precise, using the Itô interpretation of stochastic integration. Let W be an \mathbb{R}^m -valued standard unit Brownian motion (covariance I on \mathbb{R}^m), defined on time interval \mathbb{R}^+ and constructed on a probability space $(\Omega, \mathcal{F}, \mathbb{P})$. Let $\mathcal{F}_t \subset \mathcal{F}$ denote the sub- σ -algebra defined by information only on $[0, t]$, let $b \geq a \geq 0$, and let $\mathcal{M}^2([a, b]; \mathbb{R}^{d \times m})$ denote the space of all measurable random functions $r : \mathbb{R}^+ \times \Omega \rightarrow \mathbb{R}^{d \times m}$ with the property that $r(t; \cdot)$ is \mathcal{F}_t -measurable, and hence depends only on the Brownian motion on $[0, t]$, and such that

$$\mathbb{E} \int_a^b \|r(t, \omega)\|_{\text{F}}^2 dt < \infty;$$

here $\|\cdot\|_{\text{F}}$ denotes the Frobenius norm. With this notation established, we have the following properties for (6.7).

Lemma 6.1. (Properties of the Itô Integral)

For $r \in \mathcal{M}^2([a, b]; \mathbb{R}^{d \times m})$, and using the shorthand $r(t) := r(t, \omega)$, we have, for SI given by (6.7) the following:

1. (i) SI is \mathcal{F}_b -measurable and is linear in r ;
2. (ii) $\mathbb{E}(\text{SI}) = 0$;
3. (iii) $\mathbb{E}|\text{SI}|^2 = \mathbb{E} \int_a^b \|r(t)\|_{\mathbb{F}}^2 dt$.

Having defined the stochastic integral, we can now move on to SDEs. In the following, $g : \mathbb{R}^d \times \mathbb{R}^+ \rightarrow \mathbb{R}^d$ is a smooth vector-valued function, and $\gamma : \mathbb{R}^d \times \mathbb{R}^+ \rightarrow \mathbb{R}^{d \times m}$ is a smooth matrix-valued function. We consider the white-noise-driven Itô SDE

$$\frac{dv}{dt} = g(v, t) + \gamma(v, t) \frac{dW}{dt}, \quad v(0) = u. \quad (6.8)$$

The precise interpretation of (6.8) is as an integral equation for $v(t) \in C(\mathbb{R}^+, \mathbb{R}^d)$:

$$v(t) = u + \int_0^t g(v(s), s) ds + \int_0^t \gamma(v(s), s) dW(s), \quad t \geq 0, \quad (6.9)$$

where the last term is a stochastic integral. Without further assumptions, existence and uniqueness of solutions to this equation may be difficult to establish. We work under a set of assumptions that is natural in many applications: we assume that there exist $\alpha, \beta \in \mathbb{R}$ such that

$$\langle g(v, t), v \rangle \leq \alpha + \beta |v|^2 \quad \forall (v, t) \in \mathbb{R}^d \times \mathbb{R}^+. \quad (6.10)$$

We observe that for both of the Lorenz models introduced later in this section, this condition is satisfied; see the discussion following equation (2.39).

Theorem 6.2. Assume that both $g : \mathbb{R}^d \times [0, T] \rightarrow \mathbb{R}^{d \times m}$ and $\gamma : \mathbb{R}^d \times [0, T] \rightarrow \mathbb{R}^{d \times d}$ are measurable, that g satisfies (6.10), that $g(\cdot, t)$ is locally Lipschitz, uniformly in $t \in [0, T]$, that $\gamma(\cdot, t)$ is globally Lipschitz on \mathbb{R}^d , uniformly in $t \in [0, T]$, and that $v(0) = u$ is a random variable, independent of the Brownian motion $W(t)$, and satisfying

$$\mathbb{E}|u|^2 < \infty.$$

Then the SDE (6.8) has a unique solution $v \in C([0, T]; \mathbb{R}^d)$ with

$$\mathbb{E} \left(\int_0^T |v(t)|^2 dt \right) < \infty.$$

Furthermore, the solution of the SDE does not explode on $[0, T]$, almost surely.

The equation (6.8) defines a Markov process whose properties we now study in more detail. Given the function $\gamma(x, t)$ in the SDE (6.8), we define the *diffusivity matrix* $\Gamma : \mathbb{R}^d \times \mathbb{R}^+ \mapsto \mathbb{R}^{d \times d}$ by

$$\Gamma(x, t) = \gamma(x, t) \gamma(x, t)^T. \quad (6.11)$$

The Markov process defined by (6.8) is characterized by the following *generator* \mathcal{L} , defined via its action on a test function $\varphi : \mathbb{R}^d \mapsto \mathbb{R}$:

$$\mathcal{L}\varphi = g \cdot \nabla \varphi + \frac{1}{2} \Gamma : \nabla \nabla \varphi. \quad (6.12)$$

The generator plays a central role in computing the rate of change of functions of the solution of the SDE via the Itô formula.

Lemma 6.3. (Itô Formula) *Assume that the conditions of Theorem 6.2 hold. Let $v(t)$ solve (6.8) and let $\varphi \in C^{2,1}(\mathbb{R}^d \times [0, T], \mathbb{R})$. Then the process $\varphi(v(t), t)$ satisfies, for $0 \leq t \leq T$,*

$$\begin{aligned} \varphi(v(t), t) &= \varphi(u, 0) + \int_0^t \left(\frac{\partial \varphi}{\partial t}(v(s), s) + \mathcal{L}\varphi(v(s), s) \right) ds \\ &\quad + \int_0^t \langle \nabla \varphi(v(s), s), \gamma(v(s), s) dW(s) \rangle. \end{aligned}$$

Assume, furthermore, that the SDE (6.8) has coefficients f, γ that are independent of time t and that φ too is chosen to be independent of time t . Then the Itô formula gives

$$\varphi(v(t)) = \varphi(u) + \int_0^t \mathcal{L}\varphi(v(s)) ds + \int_0^t \langle \nabla \varphi(v(s)), \gamma(v(s)) dW(s) \rangle. \quad (6.13)$$

Deconstruction of (6.13) shows that the derivative of $\varphi(v(t))$ contains an extra term that would not be present if $W(t)$ were a smooth function of time, since the formula may be written as

$$\varphi(v(t)) = \varphi(u) + \int_0^t \frac{1}{2} \Gamma(v(s)) : \nabla \nabla \varphi(v(s)) ds + \int_0^t \langle \nabla \varphi(v(s)), dv(s) \rangle.$$

As in discrete time, we will make occasional reference to the concept of *ergodicity*. In the simplest setting, where the ergodic average is with respect to an *invariant measure* μ_∞ with Lebesgue density ρ_∞ , we obtain, for μ_∞ , almost every $v(0) = u$, and some class of test functions $\varphi : \mathbb{R}^d \rightarrow \mathbb{R}$,

$$\frac{1}{T} \int_0^T \varphi(v(t)) dt \rightarrow \int_{\mathbb{R}^d} \rho_\infty(v) \varphi(v) dv. \quad (6.14)$$

Expressed in terms of the measure rather than its density, we obtain

$$\frac{1}{T} \int_0^T \varphi(v(t)) dt \rightarrow \int_{\mathbb{R}^d} \varphi(v) \mu_\infty(dv). \quad (6.15)$$

In this context, it is important to define the L^2 -adjoint of \mathcal{L} , namely the operator \mathcal{L}^* given by

$$\mathcal{L}^* \varphi = -\nabla \cdot (g\varphi) + \frac{1}{2} \nabla \cdot \nabla \cdot (\Gamma\varphi). \quad (6.16)$$

When the random variable $v(t)$ solving (6.8) has a density $\rho(\cdot, t)$ with respect to Lebesgue measure on \mathbb{R}^d , then ρ solves the **Fokker–Planck** equation

$$\frac{d\rho}{dt} = \mathcal{L}^* \rho. \quad (6.17)$$

When the SDE with generator \mathcal{L} is ergodic and has an invariant measure with Lebesgue density ρ_∞ , then this density will be the unique, up to normalization, nonnegative solution of $\mathcal{L}^* \rho_\infty = 0$ in $L^1(\mathbb{R})$; in other words, the unique, up to normalization, nonnegative steady-state solution of the Fokker-Planck equation in $L^1(\mathbb{R})$. In this regard, it is useful to note that if ρ_∞ satisfies

$$-g\rho_\infty + \frac{1}{2}\nabla \cdot (\Gamma\rho_\infty) = 0 \quad (6.18)$$

and is a positive C^2 function in $L^1(\mathbb{R})$, then it also satisfies $\mathcal{L}^* \rho_\infty = 0$ and is the density of an invariant measure for the SDE; in this situation, the SDE is known as *reversible*. Furthermore, if $\gamma = \sqrt{2\epsilon}I$, so that $\Gamma = 2\epsilon I$, note that (6.18) reduces to the equation

$$-g\rho_\infty + \epsilon\nabla\rho_\infty = 0. \quad (6.19)$$

This equation has an exact solution in the case $g(\cdot) = -\nabla V(\cdot)$, namely

$$\rho_\infty(\cdot) \propto \exp(-\epsilon^{-1}V(\cdot)). \quad (6.20)$$

As in discrete time, a good way of visualizing ergodicity is via the *empirical measure*, or *histogram*, generated by a trajectory of the dynamical system. Equation (6.14) formalizes the idea that the histogram of the stochastic dynamical system defined by (6.8) converges, in the large- T limit, to the probability density function ρ_∞ of a random variable, independently of the starting point $v(0) = u$. As in discrete time, thinking in terms of pdfs of the signal, or functions of the signal, and neglecting time-ordering information is a very useful viewpoint throughout this book.

6.2 Guiding Examples

In this section, we describe various examples of the underlying signal dynamics as governed by equation (6.4). These will be used to guide and illustrate our subsequent analyses. Since we do not consider observations in this section, we refer to the driving Brownian motion simply as B , not B_v . The first two examples concern the linear problem

$$\frac{dv}{dt} + \Lambda v = \sqrt{\Sigma_0} \frac{dB_v}{dt}, \quad (6.21a)$$

$$v(0) = v_0, \quad (6.21b)$$

which generates a Gaussian process for fixed v_0 , or for v_0 itself Gaussian.

Example 6.4. Our first class of examples concerns equation (6.21) in the case $n = 2$, $\Sigma_0 = I$, and

$$\Lambda = \begin{pmatrix} 1 & 0 \\ 0 & -10^{-1} \end{pmatrix}.$$

The two components are independent in this case. Their behaviors are very different because the matrix Λ induces growth in the second component, and damping in the first. These effects must be considered in the context of the noise driving the equation, resulting in the typical properties (with respect to the random noise) that $v_1(t)$ spends most of its time fluctuating around 0, while $|v_2(t)|$ grows to infinity. This is illustrated in Figure 6.1. ♠

Example 6.5. Of particular interest in the linear case (6.21) is the situation in which $\Lambda > 0$ and $\Sigma_0 = 2I$; in this case, the solution of the resulting linear Gaussian SDE is known as an Ornstein–Uhlenbeck (OU) process. The solution may be written in integral form as

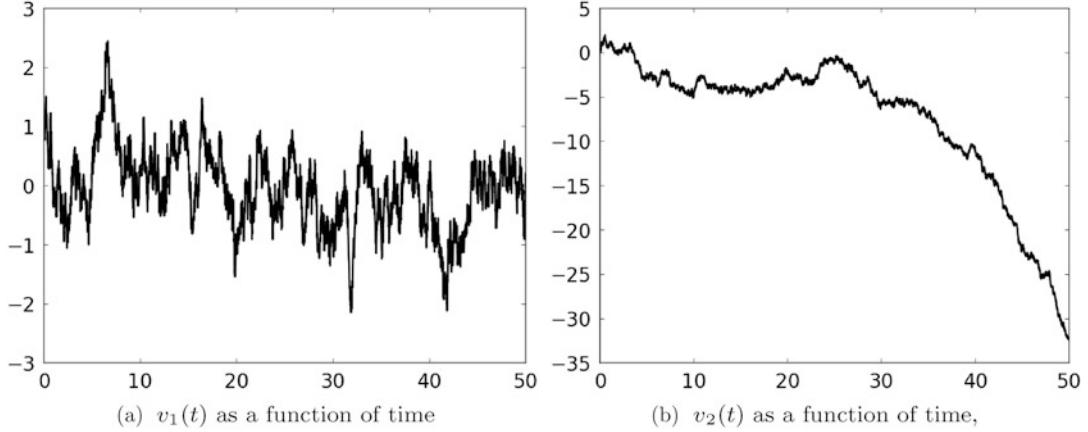


Fig. 6.1: Behavior of (6.4) for Example 6.4, with (6.4) solved with an Euler–Maruyama method ($\tau = 10^{-2}$).

$$v(t) = e^{-\Lambda t}v(0) + \sqrt{2} \int_0^t e^{-\Lambda(t-s)} dB(s).$$

From this, it is clear that the distance between any two solutions driven by the same Brownian motion, say W^* , but starting at different points, will converge exponentially fast toward each other and indeed exponentially fast towards the distinguished solution

$$v^*(t) := \sqrt{2} \int_0^t e^{-\Lambda(t-s)} dW^*(s).$$

As a linear transformation of a Brownian motion, v^* is itself Gaussian, and a straightforward calculation, using the **Itô isometry** (Lemma 6.1 (iii)) for the covariance, shows that it has mean zero and covariance $\Lambda^{-1}(I - e^{-2\Lambda t})$. In particular, the variance tends to the limit Λ^{-1} as $t \rightarrow \infty$. The OU process is in fact ergodic with unique invariant measure given by the Gaussian $N(0, \Lambda^{-1})$. This can be seen by noting, from (6.20), that every function proportional to $\exp(-\frac{1}{2}|\Lambda^{\frac{1}{2}}v|^2)$ satisfies (6.19).

Figure 6.2a shows two trajectories of this process starting from different initial conditions, but driven by the same Brownian motion B^* , in the case $n = 1$ and $\Lambda = 1$. Notice that the two trajectories converge toward each other as predicted by the analysis. Furthermore the empirical measure of each trajectory, its histogram, converges toward the unit Gaussian $N(0, 1)$ as implied by ergodicity (see Figure 6.2b). ♠

Example 6.6. Our third example concerns noise-driven motion in a double-well potential. In particular, if we consider the equation

$$\frac{dv}{dt} = -V'(v) + \sqrt{2\epsilon} \frac{dB}{dt}$$

for the potential $V(v) = \frac{1}{4}(1 - v^2)^2$, we obtain the equation

$$\frac{dv}{dt} = v - v^3 + \sqrt{2\epsilon} \frac{dB}{dt}, \quad (6.22)$$

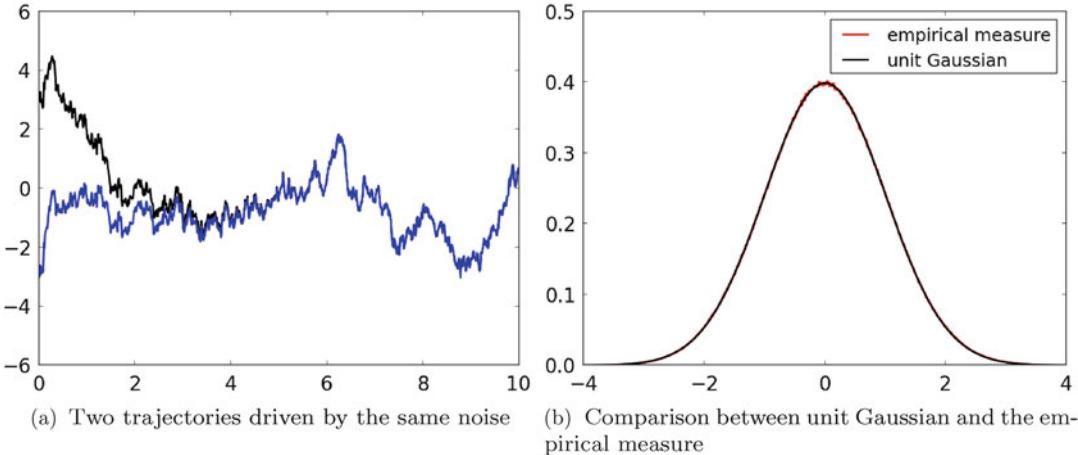


Fig. 6.2: Behavior of two different trajectories driven by the same noise of (6.4) for Example 6.4, and comparison of the empirical measure calculated from one trajectory with the ergodic measure. Empirical measure constructed by solving (6.4) with Euler–Maruyama with $\tau = 10^{-2}$ with final time of integration $T = 10^5$.

which we will use in several illustrations in the sequel. In the absence of noise, when $\epsilon = 0$, the equation will give solutions that converge to the stable equilibrium points at ± 1 , according to whether $\text{sgn}(v(0)) = \pm 1$, and will stay at the unstable equilibrium point at the origin if $v(0) = 0$. With noise ($\epsilon > 0$), the solution can make transitions between the two minima of the potential, and at random times. This leads to an ergodic Markov process that has invariant measure with Lebesgue density proportional to $\exp(-\epsilon^{-1}V(v))$, as shown in the discussion preceding (6.20); note that such a function satisfies (6.19) and reflects the fact that solutions spend most of their time near ± 1 when ϵ is small.

Figure 6.3a shows a trajectory of this SDE for $\epsilon = 0.08$, exhibiting transitions between the two stable equilibria, followed by fluctuations about them, caused by the noise. Furthermore, Figure 6.3b shows the empirical measure of the trajectory over the longer time interval $T = 5 \times 10^5$, in comparison with the density of the invariant measure, illustrating ergodicity. ♠

Example 6.7. We will also consider the Lorenz '63 model from Example 2.6, with additive white noise. We thus obtain the SDEs¹

$$\frac{dv_1}{dt} = \alpha(v_2 - v_1) + \sigma_1 \frac{dB_1}{dt}, \quad (6.23a)$$

$$\frac{dv_2}{dt} = -\alpha v_1 - v_2 - v_1 v_3 + \sigma_2 \frac{dB_2}{dt}, \quad (6.23b)$$

$$\frac{dv_3}{dt} = v_1 v_2 - b v_3 - b(r + \alpha) + \sigma_3 \frac{dB_3}{dt}, \quad (6.23c)$$

where the B_j are Brownian motions, assumed to be independent.

Figure 6.4 shows the behavior of a trajectory computed with $\sigma_1 = \sigma_2 = \sigma_3 = \sqrt{2 \times 0.25}$ and projected onto the v_1/v_2 -coordinates and onto the v_2/v_3 -coordinates; it should be compared

¹ Here the index denotes components of the solution, not discrete time.

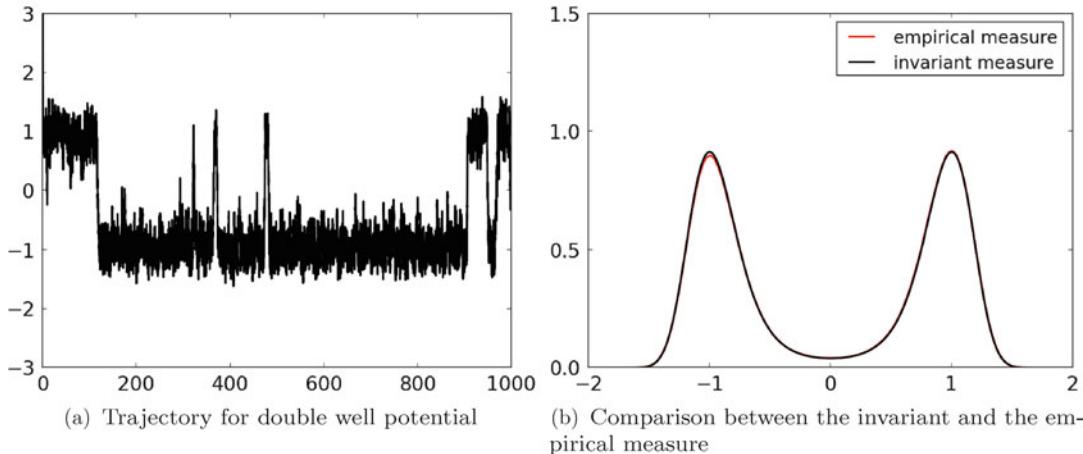


Fig. 6.3: Behavior of a trajectory of (6.4) for Example 6.6, and comparison of the empirical measure calculated from this trajectory with the invariant measure. Empirical measure constructed by solving (6.4) with Euler–Maruyama with $\tau = 10^{-2}$ with final time of integration $T = 5 \times 10^5$; see also `p1c.m` in Section 9.1.1.

with Figure 2.6, arising in the deterministic case (the scales differ slightly). Notice the similar structure to the deterministic case, but with a smearing effect caused by the noise. This problem is also ergodic, and Figure 6.5 shows the density of the invariant measure when projected onto the v_1 - and v_2 -coordinates respectively. The invariant measure is calculated from the empirical measure, using the Euler–Maruyama scheme, with $\tau = 10^{-3}$ and final time of integration $T = 10^5$. ♠

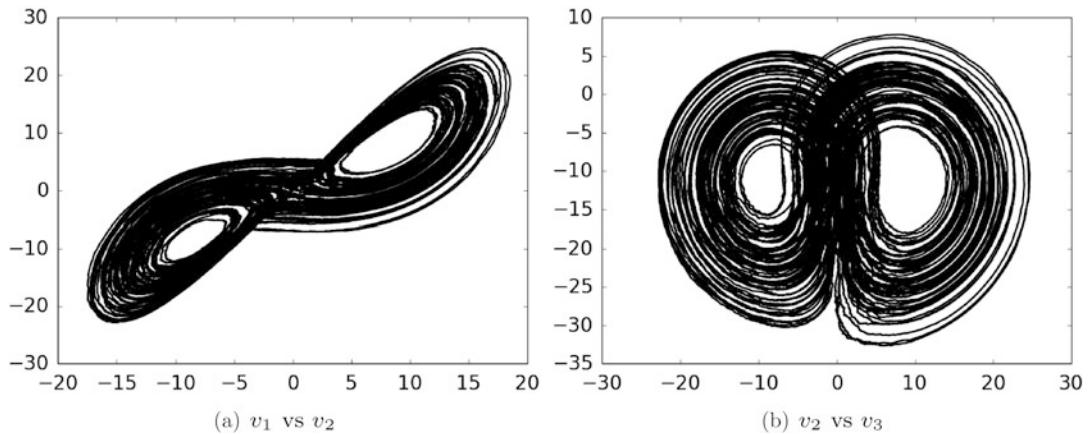


Fig. 6.4: The Lorenz equations with additive noise (6.23); projection onto two different pairs of coordinates.

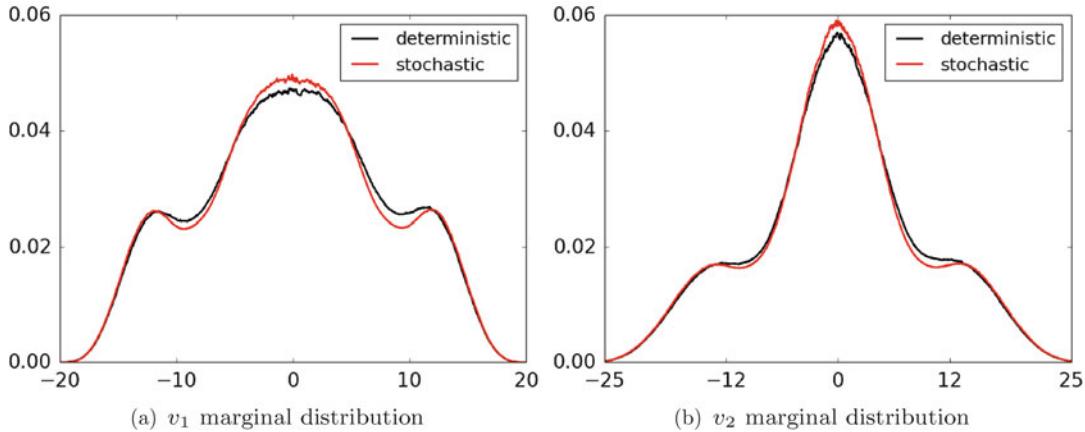


Fig. 6.5: Different marginal distributions of the invariant measure for the Lorenz equations (6.23) with additive noise. The invariant measure is also plotted in the case of zero noise (deterministic) for reference.

Example 6.8. We also consider a noisy Lorenz '96 model in the following form:²

$$\frac{dv_k}{dt} = v_{k-1}(v_{k+1} - v_{k-2}) - v_k + F + \sigma_k \frac{dB_k}{dt}, \quad k \in \{1, \dots, J\}, \quad (6.24a)$$

$$v_0 = v_J, \quad v_{J+1} = v_1, \quad v_{-1} = v_{J-1}. \quad (6.24b)$$

Again the B_k are assumed to be independent Brownian motions. This is the model from Example 2.7 extended to include additive noise. Figure 6.6 shows two different projections of (6.24) for $\sigma_k = \sigma = \sqrt{0.5}$ calculated using the Euler–Maruyama scheme for $\tau = 10^{-3}$ and final time of integration $T = 10^2$; this figure should be compared with Figure 2.9, which corresponds to the case $\sigma_k \equiv 0$. ♠

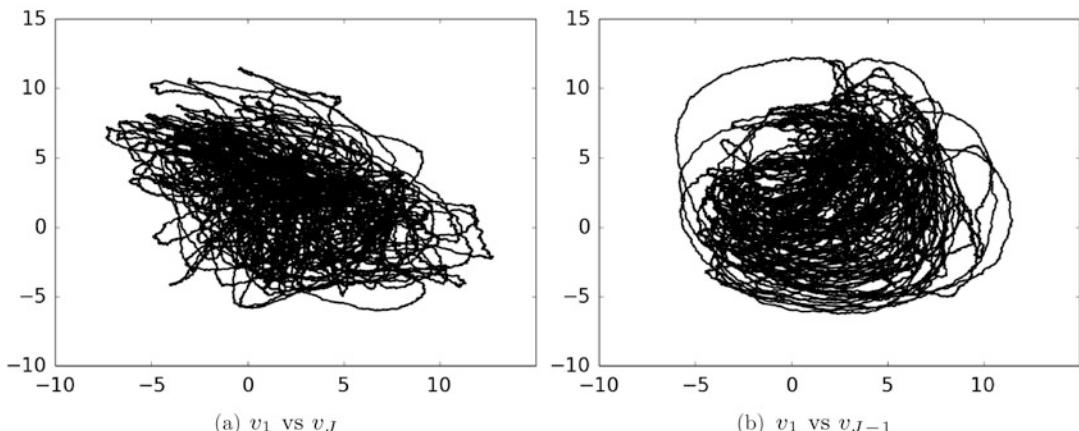


Fig. 6.6: The Lorenz '96 model with additive noise (6.24); projection onto two different pairs of coordinates.

² Again, the index here denotes components of the solution, not discrete time.

6.3 Smoothing Problem

6.3.1. Probabilistic Formulation of Data Assimilation

As in the discrete case, we consider an underlying Bayesian formulation of the data-assimilation problem in which the object of interest is the probability distribution on the signal v from (6.4) given the data z from (6.5). And in the case of deterministic dynamics (6.6), this becomes a probability distribution on the initial condition $v(0) = v_0$ given z . In both cases, we have a jointly varying random variable (u, z) , with u either the function v or, in the case of deterministic dynamics, the initial condition v_0 . The **smoothing distribution** is the conditional probability distribution $u|z$. Once again, Bayes's theorem will provide the way to determine this distribution. However, in order to derive an expression for the relevant probability distributions before conditioning, a key ingredient will be the **Girsanov formula**, which describes how to find the change of measure between two SDEs with the same diffusion coefficient. We now describe this, before turning to conditioning in the following subsection. We also emphasize that in the continuous-time setting considered here, the data, namely z , is the integral of the data y , which is employed in discrete time.

6.3.2. Girsanov Formula

SDEs of the form (6.8) generate solutions that are continuous functions of time—see Theorem 6.2—and we refer to the space $C([0, T]; \mathbb{R}^d)$ in which such solutions lie as *pathspace*. SDEs that do not have the same diffusion coefficient generate measures that are mutually singular on pathspace; the same is true of SDEs starting from different deterministic initial conditions. However, if these two possibilities are ruled out, then under conditions on the drift and diffusion coefficient, such as those specified below in Theorem 6.9, the two different SDEs generate measures that are absolutely continuous with respect to each other. The Girsanov formula provides an explicit expression for the Radon–Nikodym derivative between two such measures.

Consider the SDE

$$\frac{dv}{dt} = g(v) + \gamma(v) \frac{dW}{dt}, \quad v(0) = v_0, \quad (6.25)$$

and the same equation with the function g set to zero, namely

$$\frac{dv}{dt} = \gamma(v) \frac{dW}{dt}, \quad v(0) = v_0. \quad (6.26)$$

We assume that the unknown $v(t)$ is in \mathbb{R}^p and that the Brownian motion is also in \mathbb{R}^p . We also assume that $g : \mathbb{R}^p \rightarrow \mathbb{R}^p$ and $\gamma : \mathbb{R}^p \rightarrow \mathbb{R}^{p \times p}$ are Lipschitz on bounded sets, and that $\gamma(v)$ is invertible for every $v \in \mathbb{R}^p$. We view the solution of the equations (6.25) and (6.26) as generating measures on $H := L^2([0, T]; \mathbb{R}^p)$. The Girsanov formula is as follows.

Theorem 6.9. (Girsanov Formula) *Assume that γ and g are such that both equations (6.25) and (6.26) have solutions on $t \in [0, T]$ that do not explode almost surely. Then the measures χ and χ_0 on H generated by the two equations (6.25) and (6.26) respectively are equivalent with Radon–Nikodym derivative*

$$\frac{d\chi}{d\chi_0}(v) = \exp\left(-\int_0^T \left(\frac{1}{2}|\gamma(v)^{-1}g(v)|^2 dt - \langle \gamma(v)^{-1}g(v), \gamma(v)^{-1}dv \rangle\right)\right).$$

Remark 6.10. We find it convenient to work with measures defined on Hilbert space, primarily because some of the algorithms we describe in the next chapter exploit Gaussian structures that are most easily explained in the Hilbert space setting. However, although our setting is then the pathspace H , the solutions generated by the probability measures χ and χ_0 concentrate on the Banach space $B := C([0, T]; \mathbb{R}^n)$ with probability 1; this fact is sometimes written succinctly as $\chi(B) = \chi_0(B) = 1$. ♠

Example 6.11. Consider the two SDEs

$$\begin{aligned}\frac{dv}{dt} &= -v + \frac{dW}{dt}, \quad v(0) = v_0, \\ \frac{dv}{dt} &= \frac{dW}{dt}, \quad v(0) = v_0.\end{aligned}$$

The two Gaussian measures generated by these SDEs, the OU process and Brownian motion, are equivalent. If we denote OU measure by χ and Wiener measure (on the Brownian motion) by χ_0 , then Theorem 6.9 gives

$$\frac{d\chi}{d\chi_0}(v) = \exp\left(-\int_0^T \frac{1}{2}(|v|^2 dt + v dv)\right).$$

By the Itô formula, Lemma 6.3, we have, under χ_0 ,

$$\frac{1}{2}v^2(T) = \frac{1}{2}v^2(0) + \int_0^T v dv + \frac{1}{2}T.$$

Thus

$$\begin{aligned}\frac{d\chi}{d\chi_0}(v) &= \exp\left(-\int_0^T \frac{1}{2}|v|^2 dt - \frac{1}{2}v^2(T) + \frac{1}{2}v_0^2 + \frac{1}{2}T\right) \\ &\propto \exp\left(-\int_0^T \frac{1}{2}|v|^2 dt - \frac{1}{2}v^2(T)\right).\end{aligned}$$

We observe in passing that although we view χ and χ_0 as measures on Hilbert space H , they in fact satisfy $\chi_0(C([0, T]; \mathbb{R})) = \chi(C([0, T]; \mathbb{R})) = 1$, so that functions drawn from χ_0 and χ are almost surely continuous; thus the pointwise evaluation $v(T)$ makes sense almost surely. ♠

6.3.3. Conditional Probabilities

In the infinite-dimensional setting that arises in conditioning v on z for equations (6.4), (6.5), or $v(0)$ on z in the case $\Sigma_0 = 0$, the following **conditioning theorem** will be of paramount importance. It plays the same role that Bayes's formula (1.7) plays in the finite-dimensional setting.

Theorem 6.12. Let ς, v be probability measures on $S \times T$, where (S, \mathbb{A}) and (T, \mathbb{B}) are measurable spaces. Denote by (x, y) with $x \in S$ and $y \in T$ an element of $S \times T$. Assume that $\varsigma \ll v$ and

that ς has Radon–Nikodym derivative ϕ with respect to v , so that $\varsigma(dx, dy) = \phi(x, y)v(dx, dy)$. Assume further that the conditional distribution of $x|y$ under v , denoted by $v^y(dx)$, exists. Then the conditional distribution of $x|y$ under ς , denoted by $\varsigma^y(dx)$, exists, and $\varsigma^y \ll v^y$. The Radon–Nikodym derivative is given by

$$\frac{d\varsigma^y}{dv^y}(x) = \begin{cases} \frac{1}{c(y)}\phi(x, y), & \text{if } c(y) > 0, \text{ and} \\ 1 & \text{else} \end{cases} \quad (6.27)$$

with $c(y) = \int_S \phi(x, y) dv^y(x)$ for all $y \in T$.

6.3.4. Stochastic Dynamics

We are now ready to turn to the derivation of a formula for the posterior distribution on v solving (6.4), given the solution z of (6.5). It will be useful in what follows to consider the equation

$$\frac{dv}{dt} = \sqrt{\Sigma_0} \frac{dB_v}{dt}, \quad (6.28a)$$

$$v(0) \sim N(m_0, C_0), \quad (6.28b)$$

found from (6.4) by setting $f \equiv 0$, and the equation

$$\frac{dz}{dt} = \sqrt{T_0} \frac{dB_z}{dt}, \quad (6.29a)$$

$$z(0) = 0, \quad (6.29b)$$

found from (6.5) by setting $h \equiv 0$. Furthermore, it will also be helpful to define

$$\Xi_1(v) := \frac{1}{2} \int_0^T |f(v(t))|_{\Sigma_0}^2 dt - \int_0^T \langle f(v(t)), dv(t) \rangle_{\Sigma_0}, \quad (6.30a)$$

$$\Xi_2(v, z) := \frac{1}{2} \int_0^T |h(v(t))|_{T_0}^2 dt - \int_0^T \langle h(v(t)), dz(t) \rangle_{T_0}, \quad (6.30b)$$

$H_1 := L^2([0, T]; \mathbb{R}^n)$, $H_2 := L^2([0, T]; \mathbb{R}^m)$, and $H = H_1 \times H_2$. Let $v = \{v(t)\}_{t \in [0, T]} \in H_1$ and $z = \{z(t)\}_{t \in [0, T]} \in H_2$. The smoothing problem in the continuous case is to study the signal v given the accumulated data up to time T , namely z , more precisely, to study the random variable $v|z$ on H_1 . We refer to $[0, T]$ as the data-assimilation window. We have the following theorem:

Theorem 6.13. *Let μ_0 denote the prior measure on the random variable $v \in H_1$ defined by (6.4), and let μ denote the posterior measure for the random variable $v|z \in H_1$ with z given by (6.5). Then if f, h are bounded Lipschitz functions, it follows that $\mu \ll \mu_0$ and*

$$\frac{d\mu}{d\mu_0}(v) \propto \exp(-\Xi_2(v, z)). \quad (6.31)$$

Proof Let ψ denote the measure on H_2 found from (6.5) with v fixed, and let ψ_0 denote the measure on H_2 found from (6.29); thus ψ_0 denotes Wiener measure on H_2 . By the Girsanov formula of Theorem 6.9, we have

$$\frac{d\psi}{d\psi_0}(z) = \exp(-\Xi_2(v, z)).$$

Now let ν denote the measure on H defined by the SDEs (6.4), (6.5), and ν_0 the measure, also on H , arising in the case $h \equiv 0$. By conditioning on fixed v and then multiplying by the measure μ_0 on H_1 determined by (6.4), we find that

$$\frac{d\nu}{d\nu_0}(v, z) = \exp(-\Xi_2(v, z)).$$

Furthermore, $\nu_0(dv, dz) = \mu_0(dv) \otimes \psi_0(dz)$. Since f and h are bounded, we deduce that

$$\int \exp(-\Xi_2(v, z)) \mu_0(dv) > 0$$

almost surely. Thus Theorem 6.12 gives the desired result. \square

Remark 6.14. • This is the analogue of Theorem 2.8 in the discrete setting. Formally, Theorem 6.13, and its form in (6.32) below, can be derived from Theorem 2.8 using the scalings (6.1) and passing to the limit $\tau \rightarrow 0$. To understand this limit, it is important to notice three facts. Firstly, within Theorem 2.8, the density with respect to Lebesgue measure can be written as the product of $\exp(-J(v))$ and $\exp(-\Phi(v; y))$, and dividing through by $\exp(-J(v))$ gives an expression for the Radon–Nikodym derivative of the desired conditional measure on $v|y$ with respect to the prior measure on v governed purely by the stochastic dynamics; this expression is proportional to $\exp(-\Phi(v; y))$. Secondly, $\Phi(v; y)$ can be written, up to terms $r(y)$ that are independent of v and depend only on y , as

$$\Phi(v; y) = \sum_{j=0}^{J-1} \frac{1}{2} |\Gamma^{-\frac{1}{2}} h(v_{j+1})|^2 - \sum_{j=0}^{J-1} \langle \Gamma^{-\frac{1}{2}} y_{j+1}, \Gamma^{-\frac{1}{2}} h(v_{j+1}) \rangle + r(y).$$

Thirdly, the term involving $r(y)$ may be absorbed into the normalization constant and can hence be ignored; the remaining two terms converge, formally, to $\Xi_2(v, z)$ under the scalings (6.1).

- We let ϑ_0 denote the measure on H_1 generated by equation (6.28); thus ϑ_0 is Wiener measure convolved with a random Gaussian initial starting point. The Girsanov theorem (Theorem 6.9) shows that

$$\frac{d\mu_0}{d\vartheta_0}(v) = \exp(-\Xi_1(v)).$$

From this identity, together with (6.31), it follows that

$$\frac{d\mu}{d\vartheta_0}(v) \propto \exp(-\Xi_1(v) - \Xi_2(v, z)). \quad (6.32)$$

This expression is useful in the application of MCMC algorithms designed specifically for sampling probability measures defined via their density with respect to a Gaussian.



6.3.5. Deterministic Dynamics

If the underlying signal dynamics is deterministic, so that we have model equation (6.6), then the resulting data-assimilation problem is one of parameter estimation in SDEs. The Gaussian distribution on $v(0) = v_0$ is the prior distribution, and our aim is to condition this on the observed data z , which is the solution of (6.5). To see how this ends up as a parameter estimation problem in SDEs, we note that the solution of (6.4) with $\Sigma_0 = 0$ is simply given by $v(t) = \Psi(v_0; t)$, where $\Psi : \mathbb{R}^n \times \mathbb{R}^+ \rightarrow \mathbb{R}^n$ is the semigroup defined in (1.10). Thus (6.5) may be written

$$\frac{dz}{dt} = h(\Psi(v_0; t)) + \sqrt{\Gamma_0} \frac{dB_z}{dt}, \quad z(0) = 0. \quad (6.33)$$

Define

$$\Xi_3(v_0, z) := \frac{1}{2} \int_0^T |h(\Psi(v_0; t))|_{\Gamma_0}^2 dt - \int_0^T \langle h(\Psi(v_0; t)), dz(t) \rangle_{\Gamma_0}. \quad (6.34)$$

Note that this is simply $\Xi_2(v, z)$ evaluated with $v(t) = \Psi(v_0; t)$.

Theorem 6.15. *The posterior smoothing distribution on $v_0|z$ for the deterministic dynamics model (6.6), (6.5) is a probability measure ν on \mathbb{R}^n with density $\mathbb{P}(v_0|z) = \varrho(v)$ proportional to $\exp(-I_{\text{det}}(v_0; z))$, where*

$$I_{\text{det}}(v_0; z) = J_{\text{det}}(v_0) + \Xi_3(v_0; z), \quad (6.35a)$$

$$J_{\text{det}}(v_0) = \frac{1}{2} |v_0 - m_0|_{C_0}^2, \quad (6.35b)$$

$$\Xi_3(v_0, z) = \frac{1}{2} \int_0^T |h(\Psi(v_0; t))|_{\Gamma_0}^2 dt - \int_0^T \langle h(\Psi(v_0; t)), dz(t) \rangle_{\Gamma_0}. \quad (6.35c)$$

Proof Let ν_0 denote the probability measure $N(m_0, C_0)$ on the initial condition v_0 , and as in the previous section, let ψ_0 denote Wiener measure on H_2 . If we define the measure on $\mathbb{R}^n \times H_2$ defined by (v_0, z) by $\eta(dv_0, dz)$, and by $\eta_0(dv_0, dz)$ the same measure when $h \equiv 0$, then conditioning on v_0 and application of Theorem 6.9 shows that

$$\frac{d\eta}{d\eta_0}(v_0, z) \propto \exp(-\Xi_3(v_0, z)).$$

Noting that $\eta_0(dv_0, dz) = \nu_0(dv_0) \otimes \psi_0(dz)$ and applying the conditioning Theorem 6.12, we deduce that $v_0|z$ is distributed according to measure ν given by

$$\frac{d\nu}{d\nu_0}(v_0) \propto \exp(-\Xi_3(v_0, z)),$$

since $c = c(z)$ is finite almost surely. The desired result follows once we note that ν_0 has density with respect to Lebesgue measure that is proportional to $\exp(-\frac{1}{2}|C_0^{-\frac{1}{2}}(v_0 - m_0)|^2)$. \square

6.4 Filtering Problem

Recall equations (6.4), (6.5) and define the accumulated data up to time t by $z^t := \{z(s)\}_{s \in [0,t]}$. The filtering problem is to find a sequential update, as t increases, of the measure μ^t , the distribution of the random variable $v(t)|z^t$. To this end, recall the adjoint generator \mathcal{L}^* given by (6.16). When it is applied to equation (6.4), we obtain

$$\mathcal{L}^* \varphi = \nabla \cdot \left(-f\varphi + \frac{1}{2}\Sigma_0 \nabla \varphi \right). \quad (6.36)$$

We have the following theorem about μ^t , in which \mathbb{E}^t denotes expectation with respect to μ^t (and hence integration with respect to the density $\rho(\cdot, t)$). It characterizes the density of the smoothing distribution (respectively unnormalized density of the smoothing distribution) as the solution of the Kushner–Stratonovich (respectively Zakai) stochastic partial differential equation (SPDE). Before reading the statement of the theorem, it is instructive to recall that in the absence of data, the probability density function for v satisfying (6.4) is given by the Fokker–Planck equation (6.17). The following theorem shows how this equation should be modified to incorporate conditioning on observed data.

Theorem 6.16. *Assume that f is globally Lipschitz, that h is linearly bounded, and that the measure μ^t governing the filtering problem for equations (6.4), (6.5) has Lebesgue density $\rho(\cdot, t) : \mathbb{R}^n \mapsto \mathbb{R}^+$ for each fixed t . Then ρ solves the Kushner–Stratonovich equation*

$$\frac{\partial \rho}{\partial t} = \mathcal{L}^* \rho + \rho \left\langle h - \mathbb{E}^t h, \frac{dz}{dt} - \mathbb{E}^t h \right\rangle_{\Gamma_0}. \quad (6.37)$$

Furthermore, $\rho(v, t) = r(v, t)/\int_{\mathbb{R}^n} r(v, t) dv$, where r solves the Zakai equation

$$\frac{\partial r}{\partial t} = \mathcal{L}^* r + r \left\langle h, \frac{dz}{dt} \right\rangle_{\Gamma_0}. \quad (6.38)$$

Remark 6.17. We note that the Zakai equation is a linear stochastic PDE, while the Kushner–Stratonovich equation is a nonlinear stochastic PDE, because evaluation of expectation under \mathbb{E}^t requires integration against ρ . In the proof sketch that follows, we begin by deriving the Zakai equation and then obtain the Kushner–Stratonovich equation from it by imposing the normalization condition that the probability density function (pdf) integrates to one. ♠

Sketch Proof As in many places in the theory of continuous-time filtering, we provide a derivation based on studying the discrete-time setting under the scalings (6.1). We give references to the rigorous derivation of the two equations in the bibliographic notes at the end of the chapter.

In the following, we invoke the scalings (6.1) and think of v_j (respectively z_j) as approximation of a process $v(t)$ (respectively $z(t)$) at time $t = j\tau$. We begin by recalling the update formula (4.28) for the filtering distribution. If we assume that μ_j has unnormalized density r_j , then equation (4.28) gives

$$r_{j+1}(v_{j+1}) = \exp \left(-\frac{\tau}{2} |h(v_{j+1})|_{\Gamma_0}^2 + \langle h(v_{j+1}), z_{j+1} - z_j \rangle_{\Gamma_0} \right) \int \mathbb{P}(v_{j+1}|dv_j) r_j(v_j). \quad (6.39)$$

We now note that under the scalings (6.1), we have (since the discrete-time stochastic dynamics approximates a limiting SDE with pdf evolution governed by (6.17))

$$\int \mathbb{P}(v_{j+1}|dv_j) r_j(v_j) \approx (e^{\mathcal{L}^* \tau} r_j)(v_{j+1}).$$

Furthermore, if we define the stochastic process q via the SDE

$$\frac{dq}{ds} = -\frac{1}{2} |h(v(s))|_{\Gamma_0}^2 + \langle h(v(s)), dz(s) \rangle_{\Gamma_0},$$

then

$$\exp\left(-\frac{\tau}{2} |h(v_{j+1})|_{\Gamma_0}^2 + \langle h(v_{j+1}), z_{j+1} - z_j \rangle_{\Gamma_0}\right) \approx \exp(q(j\tau + \tau) - q(j\tau)).$$

We define

$$M_j(t) := \exp(q(j\tau + t) - q(j\tau))$$

and note that then

$$\exp\left(-\frac{\tau}{2} |h(v_{j+1})|_{\Gamma_0}^2 + \langle h(v_{j+1}), z_{j+1} - z_j \rangle_{\Gamma_0}\right) \approx M_j(\tau).$$

Applying the Itô formula of Lemma 6.3 to $M_j(t)$ shows that

$$M_j(t) = 1 + \int_{j\tau}^{j\tau+t} M_j(s) \langle h(v(s)), dz(s) \rangle_{\Gamma_0}.$$

This rather subtle calculation reflects the exponential martingale characteristics of $M_j(t)$; it is central to the derivation we give here: more naive calculations based on the scalings (6.1) will not correctly derive the limiting equations unless they reflect the exponential martingale structure that we have built in here. Using the preceding display, we obtain the natural approximation

$$M_j(\tau) \approx 1 + M_j(0) \langle h(v_{j+1}), z_{j+1} - z_j \rangle_{\Gamma_0} = 1 + \langle h(v_{j+1}), z_{j+1} - z_j \rangle_{\Gamma_0}.$$

Furthermore, we also have the natural approximation

$$(e^{\mathcal{L}^* \tau} r)(v) \approx (1 + \tau \mathcal{L}^*) r(v).$$

Combining these two approximations within (6.39) gives

$$r_{j+1}(v_{j+1}) = (1 + \langle h(v_{j+1}), z_{j+1} - z_j \rangle_{\Gamma_0})(1 + \tau \mathcal{L}^*) r_j(v_{j+1}).$$

Rearranging and keeping terms up to order $\mathcal{O}(\tau)$, we obtain

$$r_{j+1}(v_{j+1}) - r_j(v_{j+1}) = \tau \mathcal{L}^* r_j(v_{j+1}) + r_j(v_{j+1}) \langle h(v_{j+1}), z_{j+1} - z_j \rangle_{\Gamma_0},$$

which is clearly a discretization of the Zakai equation.

To obtain the Kushner–Stratonovich equation, we apply the Itô formula to the function $\rho(v, t) = r(v, t) / \int_{\mathbb{R}^n} r(v, t) dv$. The Itô formula as defined in Lemma 6.3 applies only in finite dimensions, but it generalizes to the infinite-dimensional situation under somewhat stringent conditions. We proceed purely formally, on the assumption that the Itô formula from

Lemma 6.3 indeed applies in our infinite-dimensional setting. To this end, we need to compute the derivatives of the function V , defined by

$$V(r) = \frac{r}{\int_{\mathbb{R}^n} r(v)dv}.$$

The first derivative $DV(r)$ is defined by its action on the function φ as follows:

$$DV(r)\varphi = \frac{\varphi}{\int_{\mathbb{R}^n} r(v)dv} - \frac{r \int_{\mathbb{R}^n} \varphi(v)dv}{(\int_{\mathbb{R}^n} r(v)dv)^2}.$$

Similarly, the second derivative $D^2V(r)$ is defined by its action on pairs of functions (ϕ, φ) as follows:

$$D^2V(r)[\varphi, \phi] = -\frac{\phi \int_{\mathbb{R}^n} \varphi(v)dv}{(\int_{\mathbb{R}^n} r(v)dv)^2} - \frac{\varphi \int_{\mathbb{R}^n} \phi(v)dv}{(\int_{\mathbb{R}^n} r(v)dv)^2} + 2 \frac{r \int_{\mathbb{R}^n} \phi(v)dv \int_{\mathbb{R}^n} \varphi(v)dv}{(\int_{\mathbb{R}^n} r(v)dv)^3}.$$

Applying the Itô formula then gives

$$\frac{\partial \rho}{\partial t} = DV(r) \frac{\partial r}{\partial t} + \frac{1}{2} D^2V(r)[rh^T \Gamma_0^{-\frac{1}{2}}, rh^T \Gamma_0^{-\frac{1}{2}}]. \quad (6.40)$$

We have, for

$$\mathbf{q} = \left\langle h, \frac{dz}{dt} \right\rangle_{\Gamma_0},$$

and with \mathbb{E} denoting expectation with respect to μ^t (and hence integration with respect to the density ρ),

$$\begin{aligned} DV(r) \frac{\partial r}{\partial t} &= \frac{\mathcal{L}^* r + r\mathbf{q}}{\int_{\mathbb{R}^n} r(v)dv} - \frac{r \int_{\mathbb{R}^n} r(v)\mathbf{q}dv}{(\int_{\mathbb{R}^n} r(v)dv)^2} \\ &= \mathcal{L}^* \rho + \rho\mathbf{q} - \rho \mathbb{E}^t \mathbf{q}. \end{aligned}$$

We also have

$$\frac{1}{2} D^2V(r)[rh^T \Gamma_0^{-\frac{1}{2}}, rh^T \Gamma_0^{-\frac{1}{2}}] = -\rho \langle h, \mathbb{E}^t h \rangle_{\Gamma_0} + \rho \langle \mathbb{E}^t h, \mathbb{E}^t h \rangle_{\Gamma_0}.$$

Substituting the two preceding displays into (6.40) gives the desired Kushner–Stratonovich equation. \square

6.5 Illustrations

As in Section 2.8, in the discrete-time setting, here we focus on some simple examples where the posterior distribution may be visualized easily. For this reason, we concentrate on the case of one-dimensional deterministic dynamics, since in that case, the corresponding probability measure ν is a measure on \mathbb{R} , where the initial conditions live (see Theorem 6.15), and not on the pathspace $H_1 = L^2([0, T]; \mathbb{R}^n)$. In Chapter 7, we introduce more sophisticated sampling methods that can be used to probe the probability distributions on function spaces (infinite-

dimensional spaces), which arise from noisy dynamics in which the unknown is an element of H_1 and is hence a time-dependent function.

Figure 6.7 concerns the scalar linear problem from equation 6.21 for $\Lambda = 0.5$ and $\Sigma_0 = 0$. We employ a prior $N(4, 5)$, we assume that $h(v) = v$, and set $\Gamma_0 = \gamma_0^2$ and consider two different values of γ_0 and three different values of T , the final time of observation. The figure shows the posterior distribution in these various parameter regimes. The true value of the initial condition that underlies the data is $v_0^\dagger = 0.5$. For both $\gamma_0 = 1$ and $\gamma_0 = 0.1$, we see that as the final time T over which we observe the process increases, the posterior distribution appears to converge to a limiting distribution. However, for smaller γ_0 , the limiting distribution has much smaller variance, and it is centered closer to the true initial condition at 0.5. Both of these remarks concerning the observed data can be explained by the fact that the problem is explicitly solvable: we show that for fixed γ_0 as $T \rightarrow \infty$, the posterior has a limit, which is a Gaussian with nonzero variance. And for fixed T as $\gamma_0 \rightarrow 0$, the posterior distribution converges to a Dirac measure (Gaussian with zero variance) centered at the truth v_0^\dagger .

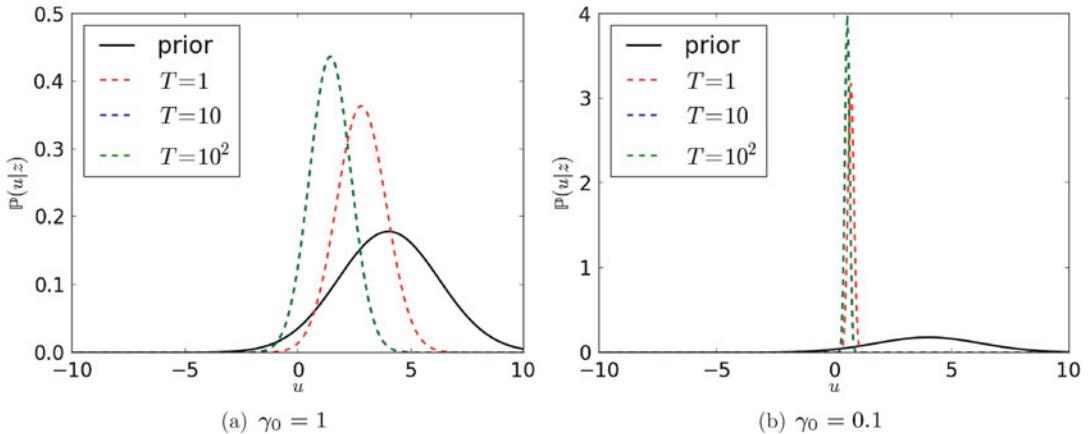


Fig. 6.7: Posterior distribution for smoothing distribution on the initial condition of equation (6.21), with $\Sigma_0 = 0$, for different levels of observational noise. The true initial condition used in both cases is $v_0^\dagger = 0.5$, while we have assumed that $C_0 = 5$ and $m_0 = 4$ for the prior distribution; see also p2c.m in Section 9.1.2.

To establish these facts, we begin by noting that from Theorem 6.15, the posterior distribution $v_0|z$ is proportional to the exponential of the negative of

$$I_{\text{det}}(v_0; z) = \frac{1}{2\sigma_0^2} |v_0 - m_0|^2 + \frac{1}{2\gamma_0^2} \int_0^T e^{-2\Lambda t} v_0^2 dt - \frac{1}{\gamma_0^2} \int_0^T e^{-2\Lambda t} v_0 v_0^\dagger dt - \frac{1}{\gamma_0} \int_0^T v_0 e^{-\Lambda t} dB_z(t),$$

where σ_0^2 denotes the prior variance C_0 , and we have used the fact that $\Psi(v_0; t) = v_0 e^{-\Lambda t}$, as well as the fact that

$$dz(t) = v_0^\dagger e^{-\Lambda t} dt + \gamma_0 dB_z(t).$$

As a quadratic form in v_0 , this defines a Gaussian posterior distribution, and we may complete the square to find the posterior mean m and variance σ_{post}^2 :

$$\frac{1}{\sigma_{\text{post}}^2} = \frac{1}{\gamma_0^2} \int_0^T e^{-2\Lambda t} dt + \frac{1}{\sigma_0^2} = \frac{1}{\gamma_0^2} \left(\frac{1 - e^{-2\Lambda T}}{2\Lambda} \right) + \frac{1}{\sigma_0^2}$$

and

$$\frac{1}{\sigma_{\text{post}}^2} m = \frac{1}{\gamma_0^2} \int_0^T v_0^\dagger e^{-2\Lambda t} dt + \frac{1}{\gamma_0} \int_0^T e^{-\Lambda t} dB_z(t) + \frac{1}{\sigma_0^2} m_0.$$

We note immediately that the posterior variance is independent of the data. Furthermore, if we fix γ_0 and let $T \rightarrow \infty$, then for every $\Lambda > 0$, we see that the large- T limit of the posterior variance is determined by

$$\frac{1}{\sigma_{\text{post}}^2} = \frac{1}{\gamma_0^2} \frac{1}{2\Lambda} + \frac{1}{\sigma_0^2}$$

and is nonzero; thus uncertainty remains in the posterior, even in the limit of infinite observational time. On the other hand, if we fix T and let $\gamma_0 \rightarrow 0$, then $\sigma_{\text{post}}^2 \rightarrow 0$, so that the uncertainty disappears in the limit. In this case, it is then natural to ask what happens to the mean. In particular, evaluating the deterministic integral in the equation for the mean gives us

$$\frac{1}{\sigma_{\text{post}}^2} m = \frac{1}{\gamma_0^2} \left(\frac{1 - e^{-2\Lambda T}}{2\Lambda} \right) v_0^\dagger + \frac{1}{\gamma_0} \int_0^T e^{-\Lambda t} dB_z(t) + \frac{1}{\sigma_0^2} m_0.$$

Using the formula for σ_{post}^2 , we obtain

$$\left(\frac{1 - e^{-2\Lambda T}}{2\Lambda} \right) m + \frac{\gamma_0^2}{\sigma_0^2} m = \left(\frac{1 - e^{-2\Lambda T}}{2\Lambda} \right) v_0^\dagger + \gamma_0 \int_0^T e^{-\Lambda t} dB_z(t) + \frac{\gamma_0^2}{\sigma_0^2} m_0.$$

From this, it follows that for fixed T and as $\gamma_0 \rightarrow 0$, we have $m \rightarrow v_0^\dagger$ almost surely with respect to the noise realization B_z . This is another example of posterior consistency.

We now study Example 6.6 in the noise-free case $\epsilon = 0$. The true dynamics are no longer linear. In particular, we are interested in studying the interplay of the underlying nonlinearity and the choice of the prior distribution. Before discussing the properties of the posterior, we draw attention to the following fact about the dynamics of the system: the system converges exponentially fast to the stable equilibrium at points ± 1 , according to whether $\text{sgn}(v_0^\dagger) = \pm 1$. This is an important observation, since it suggests that if the system is observed on long time intervals, then the posterior distribution will assign the majority of the probability to the basin of attraction of the equilibrium to which the true dynamics of the system converge.

In our case, we have chosen $v_0^\dagger = 0.5$, so the true solution converges to $+1$, which has the interval $(0, \infty)$ as its basin of attraction; we therefore expect that most of the probability in the posterior will be assigned to $(0, \infty)$. Figures 6.8 and 6.9 illustrate the fact that this is indeed the case, for large enough observation time T , for a variety of different prior choices, all Gaussian with variance C_0 . In particular, we can see that observing up to $T = 10$ is enough in order for the posterior probability to be positive primarily in the interval $(0, \infty)$. Interestingly, though, we see that choosing the prior in such a way that the majority of its probability is in the wrong basin of attraction—see Figure 6.9—leads to posteriors with quite different characteristics; in particular, they have most of their support in the true basin of attraction of $+1$ but have smaller posterior spread than those arising for the priors chosen in Figure 6.8.

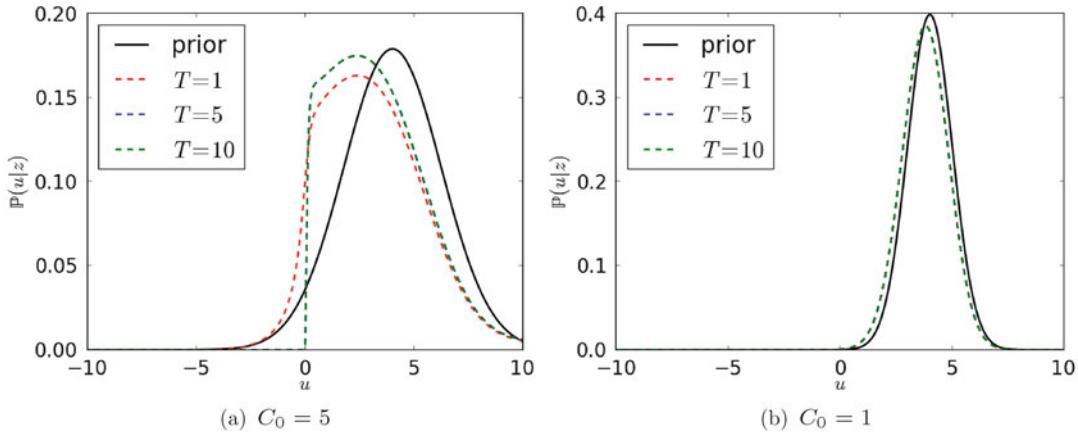


Fig. 6.8: Posterior distribution for Example 6.6, with $\epsilon = 0$, for two different prior covariances. The true initial condition used in both cases is $v_0^\dagger = 0.5$, while we have assumed $m_0 = 4$ and $\gamma_0 = 1$; see also `p3c.m` in Section 9.1.3.

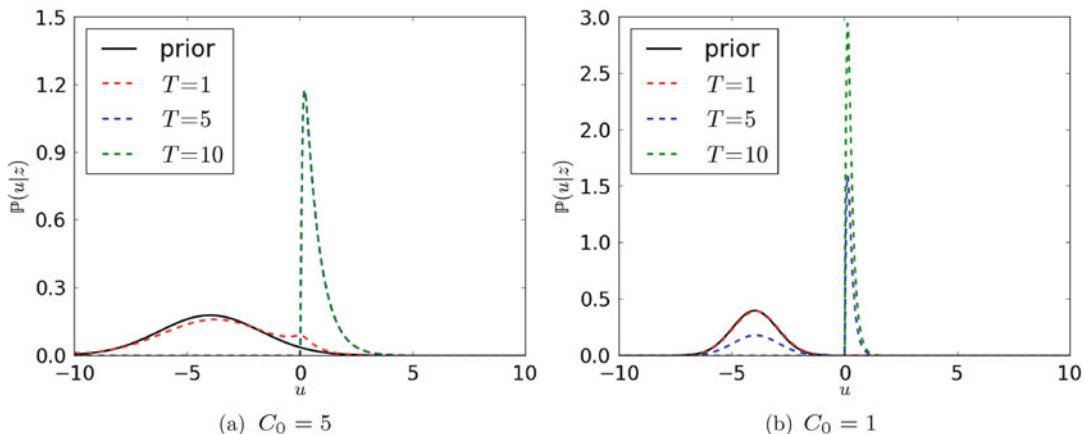


Fig. 6.9: Posterior distribution for Example 6.6, with $\epsilon = 0$, for two different prior covariances. The true initial condition used in both cases is $v_0^\dagger = 0.5$, while we have assumed $m_0 = -4$ and $\gamma_0 = 1$; see also p3c.m in Section 9.1.3.

6.6 Bibliographic Notes

- Section 6.1. In the first subsection, we give a formal derivation of the continuous-time limit of the data-assimilation setup introduced in Chapter 2. The derivations should be straightforward to understand for readers with a knowledge of the numerical solution of SDEs, as detailed in [84] or explained at an introductory level in [68]. The second section contains a brief summary of various basic results from the theory of stochastic integration and differential equations. All these results may be found in [104]: Lemma 6.1 appears as Theorem I.5.8 in that text, Theorem 6.2 as Theorem II.3.5, and Lemma 6.3 as Theorem I.6.4.
 - Section 6.2 is devoted to examples. The OU process of Example 6.5 is of fundamental importance in stochastic analysis; see section III.3.5 of [104], for example. The equation from Example 6.6 is known as the overdamped Langevin equation, or Brownian dynamics model, and is the canonical example of a reversible continuous-time Markov process; see

- [120]. Examples 6.7 and 6.8 concern noisy perturbation of a deterministic system satisfying (2.16); the ergodicity of such problems is discussed in [106] and [105].
- Section 6.3 is devoted to a definition of the smoothing problem from continuous-time data assimilation. The key Girsanov formula of Theorem 6.9 appears as Theorem XIII.2.2 in [104] in a simplified setting, while the general formulation that we state here is taken from [49], Theorem 11A. Realizations of Brownian motion are in fact Hölder continuous for every exponent up to 1/2, a property inherited by solutions of (6.25); this fact lies behind the discussion in Remark 6.10, and further discussion of regularity properties of SDEs may be found in [104, 114]. The conditioning Theorem 6.12 is from [62], Lemma 5.3. The formulation of continuous-time smoothing problems for a wide variety of models, including the one considered here, is given in [63].
 - Section 6.4 concerns the continuous-time filtering problem, a subject that is described in some detail in the book [125] and is given a modern development in [9]. In particular, the Zakai and Kushner–Stratonovich equations, derived purely formally here, are given rigorous developments in those books. To be more precise, Theorem 6.16 follows from Theorems 3.24 and 3.30 in [9], after application of Exercises 3.11 and 3.25 to simplify the conditions; note that since the initial distribution on $v(0)$ is Gaussian, all moments exist. Use of the Itô formula in infinite dimensions, as employed in our sketch derivation of the Kushner–Stratonovich equation from the Zakai equation, is discussed in [41]. Our sketch derivation is based, in part, on the very clear presentation in Chapter 4 of the lecture notes [118]. The Zakai and Kushner–Stratonovich equations can be used as the basis for numerical approximation of the filtering problem, by means of Galerkin approximation, polynomial chaos [131], and other finite-dimensional approximation techniques; see [9] and the references therein. Note, however, that these methods are not well adapted to high-dimensional problems, and new ideas are required for the numerical solution of these equations in high dimensions.
 - As in discrete time (see Section 2.5), the filtering and smoothing distributions are related. To be specific, in the case of stochastic dynamics, the measure μ given by Theorem 6.13 (which is a measure on the space H_1 defined preceding the theorem, but in fact, it is concentrated on $C([0, T]; \mathbb{R}^n)$) has marginal on the path at time T , which is a measure on \mathbb{R}^n , given by the filtering distribution μ^T from Theorem 6.16. Furthermore, in the case of deterministic dynamics, the pushforward of the measure ν given in Theorem 6.15 by T time units under the flow given by (6.4) with $\Sigma_0 = 0$ coincides with the filtering distribution μ^T from Theorem 6.16.
 - We have not included a section on the well-posedness of the data-assimilation problem, for either filtering or smoothing, because this is a considerably more subtle issue in continuous time than in discrete time. The reason for this subtle behavior is the dependence of the smoothing or the filtering distributions on a stochastic integral with respect to the observed data. In the smoothing situation, this is manifest in Theorems 6.13 and 6.15 in the stochastic and deterministic settings respectively; and in the filtering situation, it is manifest in the Kushner–Stratonovich and Zakai equations (6.37) and (6.38). The theory of rough paths provides the natural context within which to study the stability of stochastic integrals with respect to perturbation, and the paper [38] demonstrates the power of this approach in studying well-posedness of the filtering problems, as well as giving an overview of the history of the subject of well-posedness of filters. Similar rough-path ideas are likely to be successful in the study of well-posedness of the smoothing problem in continuous time.
 - We also mention that although we have not included an explicit continuous-time analogue of Section 2.7, similar considerations apply in continuous time. In particular, it is important to distinguish between the quality of the data, as manifest in the posterior distribution,

and independent of any algorithm, and the ability of the algorithm employed to represent the posterior distribution accurately.

6.7 Exercises

1. Using the Itô formula, show that

$$\int_0^T W(t) dW(t) = \frac{1}{2}W^2(T) - \frac{1}{2}T.$$

2. Consider the stochastic differential equation

$$\frac{dv}{dt} = \lambda v + \mu v \frac{dW}{dt}, \quad v(0) = v_0,$$

where $v(t) \in \mathbb{R}$ for each fixed $t \geq 0$. By applying the Itô formula to the function $f(x) = \ln x$, show that

$$v(t) = v_0 \exp\left(\left(\lambda - \frac{1}{2}\mu^2\right)t + \mu W(t)\right).$$

3. Consider the Ornstein–Uhlenbeck process and its solution from Example 6.5 with the additional assumptions that $v(t)$ is one-dimensional and that $v(0) \sim N(v_0, \sigma^2)$, independently of the driving Brownian motion $B(t)$. Using the properties of the Itô integral, calculate $\mathbb{E}(v(t))$ and $\mathbb{E}(v^2(t))$. In addition, show that up to a normalizing, constant the solution to the Fokker–Planck equation (6.17) is given by

$$\rho(v, t) \propto \exp\left(-\frac{(v - e^{-At}v_0)^2}{2(e^{-2At}\sigma^2 + A^{-1}(1 - e^{-2At}))}\right).$$

Using this formula, compute the limit of $\rho(v, t)$ as $t \rightarrow \infty$. Demonstrate that the limit distribution is invariant under the Ornstein–Uhlenbeck process. (The limiting calculation, in fact, exhibits the ergodicity of the process.)

4. Consider the differential equation

$$\frac{dv}{dt} = Av,$$

where we have $A \in \mathbb{R}^{2 \times 2}$ and the observations z are given by

$$\frac{dz}{dt} = Hv + \sqrt{\Gamma_0} \frac{dB_z}{dt}.$$

Modify the code p2c.m in order to plot the posterior on the initial condition v_0 in two dimensions in the case that the matrix A is given by

$$A_1 = \begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix}, \quad A_2 = \begin{pmatrix} \lambda & \alpha \\ 0 & \lambda \end{pmatrix}, \quad A_3 = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}$$

and the observation operator is given by

$$H_1 = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}, \quad H_2 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}.$$

Consider the cases in which λ_1 and λ_2 have the same sign and opposite signs, as well as positive and negative λ . For which of the different choices do you observe posterior consistency?

5. Consider the Lorenz '63 model given in Example 6.7. Experiment with the effect of the noise level on the marginal distribution of the invariant measure, as depicted in Figure 6.5, using program p16.m as your starting point. Consider varying the size of the noise homogeneously in each component, as well as setting the noise to zero in all but one of the three components, and then considering the effect of varying the noise in just the one component.
6. Recall the Langevin equation (6.22),

$$\frac{dv}{dt} = v - v^3 + \sqrt{2\epsilon} \frac{dW}{dt},$$

from Example 6.6. Consider the following linearly implicit discretization of the equation:

$$v_{n+1} = (1 + \tau)v_n - \tau v_{n+1} v_n^2 + \sqrt{2\epsilon\tau} \tilde{\xi}_n,$$

where $\tilde{\xi}_n$ is an independent $N(0, 1)$ random variable. On rearranging, we find that

$$v_{n+1} = (1 + \tau v_n^2)^{-1} ((1 + \tau)v_n + \sqrt{2\epsilon\tau} \tilde{\xi}_n).$$

Modify the code in program pc1.m to study the invariant measure of this equation numerically, using this linearly implicit discretization, for different noise levels ϵ .

Chapter 7

Continuous Time: Smoothing Algorithms

In this chapter, we describe various algorithms for the smoothing problem in continuous time. We begin, in Section 7.1, by describing the **Kalman–Bucy smoother**, which applies in the case of linear dynamics when the initial conditions and the observational noise are Gaussian; the explicit Kalman–Bucy formulas are useful for the building of intuition. In Section 7.2, we discuss MCMC methods to sample from the smoothing distributions of interest. However, as in the discrete-time case, sampling the posterior can be prohibitively expensive. For this reason, it is of interest to identify the point that maximizes probability, using techniques from optimization, rather than explore the entire probability distribution—the variational method. This optimization approach is discussed in Section 7.3. Section 7.4 is devoted to numerical illustrations of the methods introduced in the previous sections. The chapter concludes with bibliographic notes in Section 7.5 and exercises in Section 7.6.

7.1 Linear Gaussian Problems: The Kalman–Bucy Smoother

As in discrete time, the Kalman smoother plays an important role, because it provides an example of a smoothing distribution that can be explicitly characterized, as we will show here. We set

$$f(v) = Lv, \quad h(v) = Hv, \tag{7.1}$$

where $L \in \mathbb{R}^{n \times n}$ and $H \in \mathbb{R}^{m \times n}$. We consider the signal/observation model (6.4), (6.5), and the resulting posterior distribution $v|y$.

Theorem 7.1. *Let μ_0 be the Gaussian measure on $H_1 = L^2([0, T]; \mathbb{R}^n)$ determined by*

$$\frac{dv}{dt} = Lv + \sqrt{\Sigma_0} \frac{dB_v}{dt}, \quad v(0) \sim N(m_0, \Gamma_0), \tag{7.2}$$

and define

$$\Xi_2(v, z) := \frac{1}{2} \int_0^T |Hv(t)|_{\Gamma_0}^2 dt - \int_0^T \langle Hv(t), dz(t) \rangle_{\Gamma_0}. \tag{7.3}$$

Electronic supplementary material The online version of this chapter (doi: [10.1007/978-3-319-20325-6_7](https://doi.org/10.1007/978-3-319-20325-6_7)) contains supplementary material, which is available to authorized users.

Then the posterior smoothing distribution on $v|y$ for the linear stochastic dynamics model (6.4), (6.5), (7.1) is a measure μ that is absolutely continuous with respect to μ_0 and is characterized by

$$\frac{d\mu}{d\mu_0}(v) \propto \exp(-\Xi_2(v, z)). \quad (7.4)$$

Proof This is a straightforward application of Theorem 6.13. \square

Although the preceding result does not explicitly characterize the mean and covariance of the Gaussian measure μ , it is possible to do so, and references are given in the bibliography. We now consider the Kalman smoother in the case of deterministic dynamics. Theorem 6.15 gives the following:

Theorem 7.2. *The posterior smoothing distribution on $v_0|y$ for the deterministic linear dynamics model (6.6), (6.5), (7.1) with symmetric positive definite C_0 and Γ_0 is characterized by Gaussian measure $\nu = N(m_{\det}, C_{\det})$. The covariance C_{\det} is the inverse of the positive definite symmetric matrix L_{\det} given by the expression*

$$L_{\det} = C_0^{-1} + \int_0^T e^{L^T t} H^T \Gamma_0^{-1} H e^{L t} dt. \quad (7.5)$$

The mean m_{\det} solves

$$L_{\det} m_{\det} = C_0^{-1} m_0 + \int_0^T e^{L^T t} H^T \Gamma_0^{-1} dz(t).$$

Proof By Theorem 6.15, we know that ν has pdf proportional to $\exp(-I_{\det}(v_0; z))$, where

$$I_{\det}(v_0; z) = \frac{1}{2} |v_0 - m_0|_{C_0}^2 + \frac{1}{2} \int_0^T |H e^{L t} v_0|_{\Gamma_0}^2 dt - \int_0^T \langle H e^{L t} v_0, dz(t) \rangle_{\Gamma_0}.$$

The inverse covariance L_{\det} satisfies

$$L_{\det} = \partial_{v_0}^2 I_{\det}(v_0; z),$$

and the mean solves

$$L_{\det} m_{\det} = -\nabla_{v_0} I_{\det}(v_0; z) \Big|_{v_0=0}.$$

These two equations characterize C_{\det} and m_{\det} . Finally, the positive-definiteness of L_{\det} follows from that of C_0 and Γ_0 . Indeed, from (7.5), we have that for $v \neq 0$,

$$\langle v, L_{\det} v \rangle \geq \langle v, C_0^{-1} v \rangle > 0.$$

\square

Remark 7.3. *The formulas for the mean and covariance given in the preceding theorem may be found by imposing the scalings from (6.1) and taking the limit $\tau \rightarrow 0$ in the formulas for the mean and covariance in Theorem 3.2 (discrete time), after setting $M = \exp(L\tau)$. Furthermore, as in Theorem 3.2, the formulas for the mean may be interpreted as a maximizer of the posterior probability of v_0 given data z .* ♠

7.2 Markov Chain–Monte Carlo Methods

As in the case of discrete time, we may use MCMC methods to sample from the smoothing distributions of interest. We start with deterministic dynamics, where the posterior distribution is on the initial condition in \mathbb{R}^n ; we then move on to the case of stochastic dynamics, where the posterior distribution is on the pathspace $H_1 = L^2([0, T]; \mathbb{R}^n) \supset C([0, T]; \mathbb{R}^n)$; as observed in Remark 6.10, the solutions are in $C([0, T]; \mathbb{R}^n)$, almost surely.

7.2.1. Deterministic Dynamics

In the case of deterministic dynamics (6.6), the measure of interest is a measure on the initial condition v_0 in \mathbb{R}^n . As in the discrete case, we describe application of the **random walk Metropolis** (RWM) sampler in this setting, where we sample the initial condition, since it is perhaps the simplest algorithm to describe. Recall from Section 6.3.5 that the measure of interest is ν with pdf ϱ and that Theorem 6.15 shows that $\varrho \propto \exp(-\mathbf{l}_{\text{det}}(v_0; y))$.

The RWM method proceeds as before: given that the state of the Markov chain at step $n - 1$ is $v^{(n-1)}$, we propose

$$w^{(n)} = v^{(n-1)} + \beta \iota^{(n-1)},$$

where $\iota^{(n-1)} \sim N(0, C_{\text{prop}})$ is a centered Gaussian random variable on \mathbb{R}^n defined for some positive definite proposal covariance C_{prop} and proposal variance scale parameter β ; we then accept the move $v^{(n)} = w^{(n)}$ with probability $a(v^{(n-1)}, w^{(n)})$, where

$$\begin{aligned} a(v, w) &= 1 \wedge \frac{\varrho(w)}{\varrho(v)} \\ &= 1 \wedge \exp(\mathbf{l}_{\text{det}}(v; y) - \mathbf{l}_{\text{det}}(w; y)); \end{aligned}$$

otherwise, we set $v^{(n)} = v^{(n-1)}$. Thus moves that decrease \mathbf{l}_{det} are always accepted; moves that increase \mathbf{l}_{det} are accepted with a probability that is smaller for larger increases. Recall that \mathbf{l}_{det} is the sum of the prior penalization (background) and the model–data misfit functional given by (6.35). The algorithm thus has a very natural interpretation in terms of the data-assimilation problem. Numerical results illustrating it are given in Section 7.3. As in discrete time, we are not advocating the RWM algorithm as an algorithm that is particularly well adapted to the sampling question at hand; we are merely using it to illustrate the application of MCMC to the smoothing problem for deterministic dynamics. More sophisticated methods are now described, in the context of stochastic dynamics.

7.2.2. Stochastic Dynamics

Here we apply the Metropolis–Hastings methodology to the data-assimilation smoothing problem in the case of the stochastic dynamics model (6.4), given data determined by (6.5). Thus the probability measure is on $v \in H_1 = L^2([0, T]; \mathbb{R}^n)$. This is a fundamentally more complicated situation than we have encountered in any of our preceding discussions of MCMC methods, both in this chapter and in Chapter 3. This is because the space from which we wish to sample (pathspace) is infinite-dimensional. This has implications for the algorithms

that may be used: for example, the RWM algorithm is not defined in our infinite-dimensional setting. In practice, we discretize the infinite-dimensional space and could, in principle, use standard MCMC methods. But the fact that the methods are not defined in the infinite-dimensional limit means that they behave poorly when the dimension of the approximating space is high. For this reason, we concentrate here on special MCMC methods that are well defined in the infinite-dimensional setting. It turns out that the two methods we introduced in Section 3.2.4 for studying discrete time dynamics are well defined in the infinite-dimensional limit. We now describe them in the current infinite-dimensional setting.

Our interest is in the measure μ on the pathspace H_1 determined by Theorem 6.13. This measure is characterized in (6.31) via its density with respect to measure μ_0 ; the measure μ_0 is simply the measure on H_1 defined by solutions of the equation (6.4). On the other hand, the measure μ is also characterized in (6.32) via its density with respect to measure ϑ_0 given by (6.28); thus the measure ϑ_0 is simply the Gaussian measure on H_1 generated by Brownian motion with a Gaussian random starting point. The two methods we now describe require, respectively, the ability to sample from μ_0 and from ϑ_0 .

We begin with the **independence dynamics sampler**. Here we choose the proposal $w^{(n)}$, independently of the current state $v^{(n-1)}$, from the prior μ_0 : this just means generating an independent sample from the unconditioned dynamics defined by (6.4). Similarly to the case of discrete-time dynamics, this move is accepted with probability $a(v^{(n-1)}, w^{(n)})$, where

$$a(v, w) = 1 \wedge \exp(\Xi_2(v, z) - \Xi_2(w, z)).$$

The resulting MCMC method always accepts moves that decrease the functional $\Xi_2(\cdot; z)$, which is defined just before Theorem 6.13; if this functional increases, then the move is accepted with a probability less than 1. It is illuminating to note that if z were differentiable, we could write

$$\Xi_2(v, z) = \frac{1}{2} \int_0^T \left| h(v(t)) - \frac{dz}{dt} \right|_{\Gamma_0}^2 dt + K(z),$$

where $K(z)$ is independent of v . Thus $\Xi_2(\cdot, z)$ is a form of model–data misfit functional: $\Xi_2(v, z)$ measures how well the given function v fits the observed data, in a least-squares sense. Furthermore, if we pretend that z is differentiable, we then have that

$$\Xi_2(v, z) - \Xi_2(w, z) = \frac{1}{2} \int_0^T \left| h(v(t)) - \frac{dz}{dt} \right|_{\Gamma_0}^2 dt - \frac{1}{2} \int_0^T \left| h(w(t)) - \frac{dz}{dt} \right|_{\Gamma_0}^2 dt,$$

and the acceptance probability may be interpreted as biasing samples toward trajectories that have a best fit to the data, in a least-squares sense.

As in the case of discrete time, and as we will see in the illustrations in Section 7.4 below, the independence dynamics sampler can be quite inefficient, because the proposed moves attempt to move far from the current state based only on the underlying stochastic dynamics, and not on the observed data; this can lead to frequent rejections. As in the discrete-time setting, this effect can be controlled by making local proposals, and such a method is exemplified by the **pCN method**. Recall the Gaussian measure ϑ_0 , noting that this has mean m the constant function $m(t) = m_0$ with fluctuations described by standard Brownian motion on \mathbb{R}^n with variance Σ_0 , and starting from an initial condition distributed as $N(0, C_0)$; as such, it is straightforward to draw from this measure. The pCN method, introduced in Section 3.2.4, is based on the following proposal: we define

$$w^{(n)} = m + (1 - \beta^2)^{\frac{1}{2}} (v^{(n-1)} - m) + \beta \iota^{(n-1)},$$

where $\iota^{(n-1)}$ is a standard Brownian motion on \mathbb{R}^n with variance Σ_0 and starting from an initial condition drawn at random from $N(0, C_0)$, and is independent of $v^{(n-1)}$. This proposal preserves ϑ_0 and would be accepted all the time if $f \equiv 0$ and $h \equiv 0$. With nonzero (f, h) , so that the observations contain information about the signal, and so that the signal itself is not simply Brownian motion, the move is accepted with probability $a(v^{(n-1)}, w^{(n)})$, where

$$a(v, w) = 1 \wedge \exp(\Xi_2(v, z) - \Xi_2(w, z) + \Xi_1(v) - \Xi_1(w)),$$

and Ξ_1, Ξ_2 are as defined preceding Theorem 6.13. By choosing β small, so that $w^{(n)}$ is close to $v^{(n-1)}$, we can make $a(v^{(n-1)}, w^{(n)})$ reasonably large and obtain a usable algorithm. These two infinite-dimensional MCMC methods are illustrated in Section 7.4.

7.3 Variational Methods

Recall that the idea of variational methods, as described in the discrete-time setting, is to find a point that maximizes the posterior probability: the MAP estimator. In finite-dimensional terms, when the posterior probability has density with respect to Lebesgue measure, this point is found by maximizing the pdf. In finite dimensions it is straightforward to relate the problem of maximizing probability to that of minimizing the negative logarithm of the probability, and this is expressed in Theorems 3.10 and 3.12. In those theorems, we also express the idea of a probability maximizer in a way that translates well to infinite dimensions, where there is no Lebesgue measure. More precisely, we find points that maximize the probability of small balls centered at that point, and then consider the limit in which the small ball has radius that tends to zero. We outline application of this definition of MAP estimator to the case of stochastic dynamics. We then consider the simpler case of deterministic dynamics, where Lebesgue measure can be used.

For stochastic dynamics, our starting point is the formula (6.32) for the measure μ on $v|z$ governed by equations (6.4), (6.5), where we note that Ξ_1 and Ξ_2 are given by (6.30). The measure ϑ_0 is the Gaussian measure generated by the equation (6.28). This has mean $m \in H_1$, which is the constant function $m(t) = m_0$, and covariance defined by the Wiener measure with variance Σ_0 and random initial condition distributed according to $N(0, C_0)$. This gives a Gaussian measure $N(m, C)$ on H_1 . A useful way of characterizing C is via its inverse L , the precision. For this problem, the precision is given by

$$L = -\Sigma_0 \frac{d^2}{dt^2},$$

with domain of L specified by the boundary conditions $\frac{dv}{dt} = \Sigma_0 C_0^{-1} v$ at $t = 0$ and $\frac{dv}{dt} = 0$ at $t = T$. If we pretend for a moment that the Gaussian measure ϑ_0 has a Lebesgue density, then by analogy with the finite-dimensional setting, it should have Lebesgue density proportional to

$$\exp\left(-\frac{1}{2}|C^{-\frac{1}{2}}(v - m)|^2\right) = \exp\left(-\frac{1}{2}|L^{\frac{1}{2}}(v - m)|^2\right).$$

This suggests, based on (6.32), that the posterior measure of interest, μ , has Lebesgue density proportional to $\exp(-I(v; z))$, where

$$I(v; z) := \frac{1}{2}|L^{\frac{1}{2}}(v - m)|^2 + \Xi_2(v; z) + \Xi_1(v).$$

This, in turn, suggests that points that maximize probability are those that minimize the functional $\mathbf{I}(\cdot; z)$; minimization is over functions v such that $v - m$ is in the domain of $L^{\frac{1}{2}}$. This leads to an infinite-dimensional minimization (indeed infimization) problem; furthermore, because of the structure of L , the resulting Euler–Lagrange equations are of second order in time. These kinds of arguments can be made rigorous, but some very subtle issues in stochastic analysis arise in making them so, and these are beyond the scope of these notes—we give detailed reference to the relevant literature in Section 7.5. Here we simply note that the dependence of Ξ_1 and Ξ_2 , as defined in (6.30), on stochastic integrals with respect to v and z respectively, makes the problem particularly hard; integration by parts, using the Itô formula, can be used to derive reformulations of the objective function $\mathbf{I}(\cdot; z)$ appropriate to proving that minimizers coincide with MAP estimators.

In the case of deterministic continuous-time dynamics, the minimization is simpler, since it is over the finite-dimensional space \mathbb{R}^n : the objective functional is $\mathbf{l}_{\text{det}}(v_0; z)$ given in Theorem 6.15. We highlight some important structure in this minimization problem. We assume that the data $z(t)$ is derived from (6.5) with $v(t) = \Psi(v_0^\dagger; t)$. Then, up to an additive constant that is independent of v_0 , we have

$$\Xi_3(v_0, z) = \frac{1}{2} \int_0^T \left| (h(\Psi(v_0; t)) - h(\Psi(v_0^\dagger; t))) \right|_{\Gamma_0}^2 dt - \int_0^T \left\langle h(\Psi(v_0; t)), \sqrt{\Gamma_0} dB_z(t) \right\rangle_{\Gamma_0} + c(v_0^\dagger),$$

where B_z is the specific realization of Brownian motion that gives rise to the data: we have substituted for z using

$$\frac{dz}{dt} = h(\Psi(v_0^\dagger; t)) + \sqrt{\Gamma_0} \frac{dB_z}{dt}.$$

Of course, we do not know B_z in practice; we simply observe z . But studying the minimization in this form, in which the truth value v^\dagger of the initial condition appears explicitly, is instructive. In this form, the minimization problem for \mathbf{l}_{det} given by (6.35) is equivalent to the minimization of

$$\frac{1}{2} |v_0 - m_0|_{C_0}^2 + \frac{1}{2} \int_0^T \left| (h(\Psi(v_0; t)) - h(\Psi(v_0^\dagger; t))) \right|_{\Gamma_0}^2 dt - \int_0^T \left\langle h(\Psi(v_0; t)), \sqrt{\Sigma_0} dB_z(t) \right\rangle_{\Gamma_0},$$

where B_z is the specific realization of Brownian motion that gave rise to the data. To understand how the global minimizer may be characterized in the limit of large T , note that the first term is then $\mathcal{O}(1)$, the second term is $\mathcal{O}(T)$ (for $v_0 \neq v_0^\dagger$ and provided that $\Psi(\cdot; t)$ is not contracting), and the third term, by the Itô formula of Lemma 6.3, is $\mathcal{O}(T^{\frac{1}{2}})$. Thus, for large T , minimization will be dominated by the middle term, which is nonnegative, and zero for $v_0 = v_0^\dagger$. This suggests that for large T , the variational method will have a global minimum close to the truth. Such arguments can be made rigorous; we give references in Section 7.5.

7.4 Illustrations

We describe several numerical experiments that illustrate the application of variational methods and MCMC methods to the smoothing problems arising in continuous time for both deterministic and stochastic dynamics.

The first illustration concerns variational methods, and in particular, application of optimization techniques to minimize \mathbf{l}_{det} given in Theorem 6.15. We consider the deterministic Lorenz '63 model from Example 6.7 with $\sigma_1 = \sigma_2 = \sigma_3 = 0$ and linear observation operator $h(v) = Hv$ with $H = (1, 0, 0)$, so that only the first component of the equation is observed; we

assume that the scalar observation equation (6.5) is employed with $\Gamma_0 = \gamma^2$. The remaining parameter values for the objective functions defined in Theorem 6.15 are $T = 2$, $\gamma = 0.1$, $C_0 = 0.01I$, and $m_0 = 0$. The value of the time step is set to $\tau = 10^{-4}$. The MAP estimator in this case (which we define to be the best of several local minima that we identified) is very close to the truth. This is shown in the top left panel of Fig. 7.1, along with several local minima. These results were generated with p7c.m. The remaining three panels show the value of the objective function I_{det} from Theorem 6.15 for each of the three components of the initial condition, with the values of the other two degrees of freedom fixed at the MAP estimator. Clockwise from top right, they are $I_{\text{det}}(v_1, v_2^{\text{MAP}}, v_3^{\text{MAP}})$, $I_{\text{det}}(v_1^{\text{MAP}}, v_2, v_3)$, and $I_{\text{det}}(v_1^{\text{MAP}}, v_2, v_3^{\text{MAP}})$. Here v_1, v_2, v_3 (possibly with superscript MAP) denote the three components of the initial condition for the Lorenz equation, a notation commented on in the footnote within Example 6.7. Notice the multimodality of these three slices through the objective function, and notice broad similarities (in that the distribution is multimodal) with the example shown in Figure 3.8(b) for the discrete case. This multimodality is a hallmark of performing MAP estimation for chaotic dynamical systems, and it can cause considerable computational difficulties, especially in high-dimensional problems.

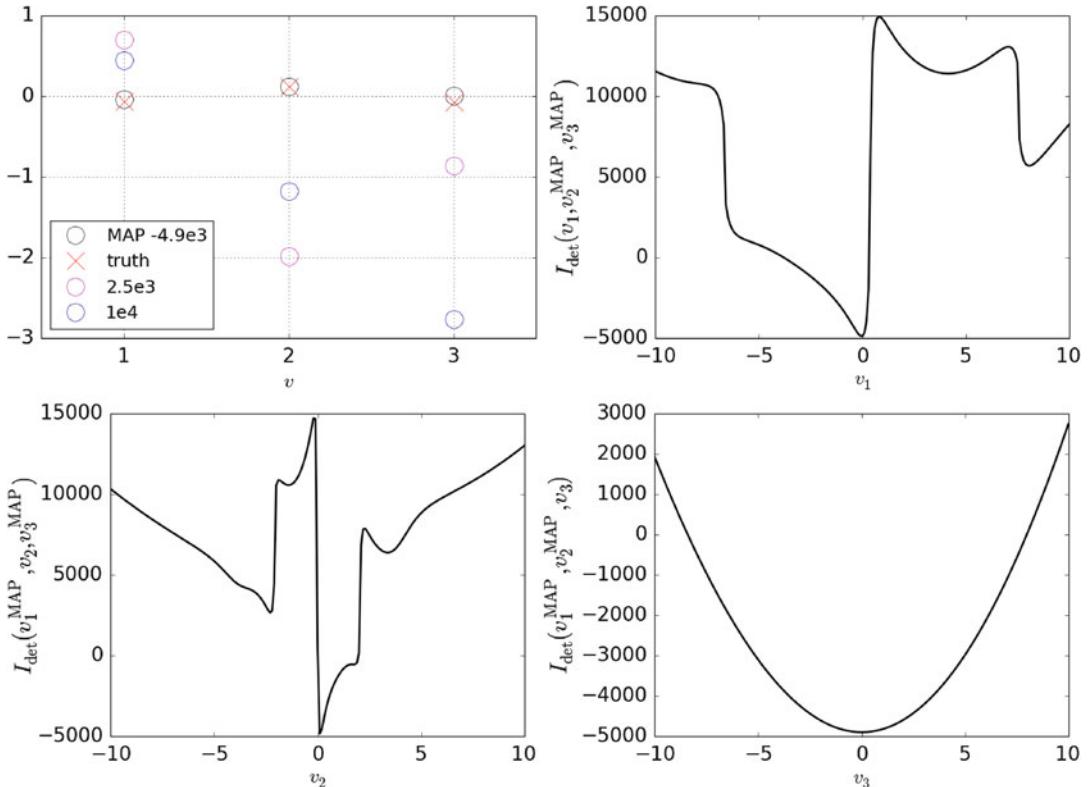


Fig. 7.1: 4DVAR for the (deterministic) Lorenz '63 equation from Example 6.7 with observation operator $H = (1, 0, 0)$. Panel (a) shows the truth, the MAP estimator (the best local minimizer that we found), and several other local minima of I_{det} ; for the three local minima, we report the value of the objective function I_{det} . Panels (b)–(d) show I_{det} as a function of each of the degrees of freedom individually, conditional on the other two being fixed at values of the MAP estimator. Recall that here v_1, v_2, v_3 denote the three components of the initial condition from Example 6.7.

The next illustration concerns the use of the RWM algorithm to study the smoothing distribution for Example 6.6. We consider the case of deterministic dynamics, with $\epsilon = 0$, so that our aim is to find the posterior distribution $\mathbb{P}(v_0|z)$ on \mathbb{R} . Recall Figure 6.8a, which shows the true posterior pdf, found by plotting the formula (6.35) given in Theorem 6.15 for the choice of Gaussian prior $N(4, 10)$, which we will also use here. We now approximate the true

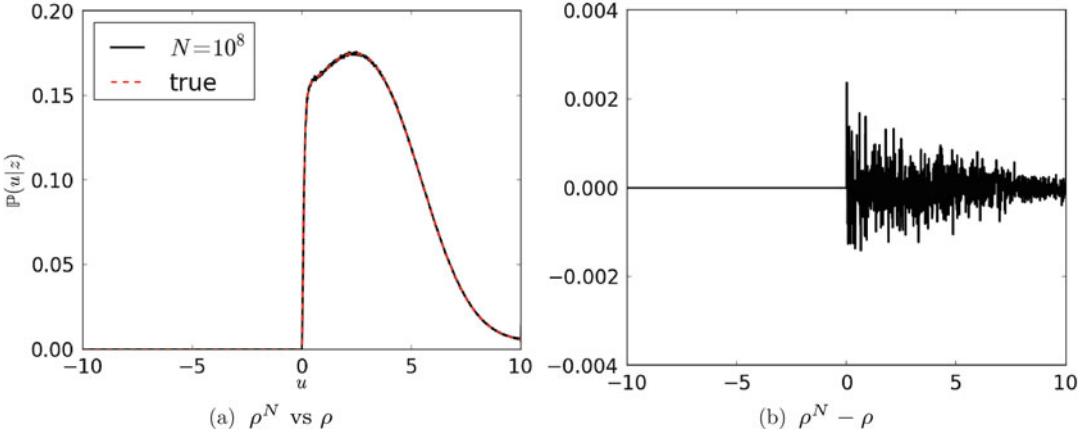


Fig. 7.2: Comparison of the posterior for Example 6.6, with $\epsilon = 0$, using random walk Metropolis and equation (6.35) directly as in the MATLAB program `p3c.m`. We have used $T = 10$, $C_0 = 5$, $m_0 = 4$, $\gamma = 1$, $\tau = 10^{-2}$, and true initial condition $v_0^\dagger = 0.5$; see also `p4c.m` in Section 9.1.3. The MCMC algorithm is run for $K = 10^8$ steps.

posterior pdf by the RWM method, using the same parameters as in Figure 6.8a, namely $T = 10$, $C_0 = 4$, $m_0 = 4$, $\gamma = 1$, $\tau = 10^{-2}$, and $v_0^\dagger = 0.5$. In Figure 7.2, we compare the posterior pdf calculated by the MCMC method (denoted by ρ^N , the histogram of the output of the Markov chain) with the true posterior pdf ρ . The two distributions are almost indistinguishable when plotted together in Figure 7.2a; in Figure 7.2b, we plot their differences, which, as we can see, is small relative to the true value. We deduce that the numbers of samples used, $K = 10^8$, results in this case in an accurate sampling of the posterior.

We now turn to the use of MCMC methods to sample the smoothing pdf $\mathbb{P}(v|z)$ in the case of stochastic dynamics (6.4). We again study Example 6.6, now with $\epsilon = 0.08$. We use both the independence dynamics sampler and the pCN method. In particular, in Figure 7.3, we plot the accept–reject ratio for the independence dynamics sampler, employing different values of observational noise γ and different observational times T . We observe that the behavior of the accept–reject ratio is similar to that in the discrete case: it decreases with decreasing γ and with larger values of the length of the time interval over which data is collected, here T (and J in discrete time). In addition, in Figure 7.4, we plot the output and the running average of the output for $u(T)$. The true value of $u(T)$ is -1.1905 . The information content of the data set is high, and the algorithm reconstructs a mean posterior value for the signal at the end time $t = T$ that is fairly close to this true value. Figure 7.4a clearly exhibits the fact that there are many rejections caused by the low acceptance probability. Figure 7.4b shows that the running average has not yet converged after 10^5 steps, indicating that the chain needs to be run for longer. If we run the Markov chain over $K = 10^6$ steps, then we get convergence, as illustrated in Figure 7.5. In particular, in Figure 7.5a, we see that the running average has converged to its limiting value when this many steps are used. Furthermore, in Figure 7.5b,

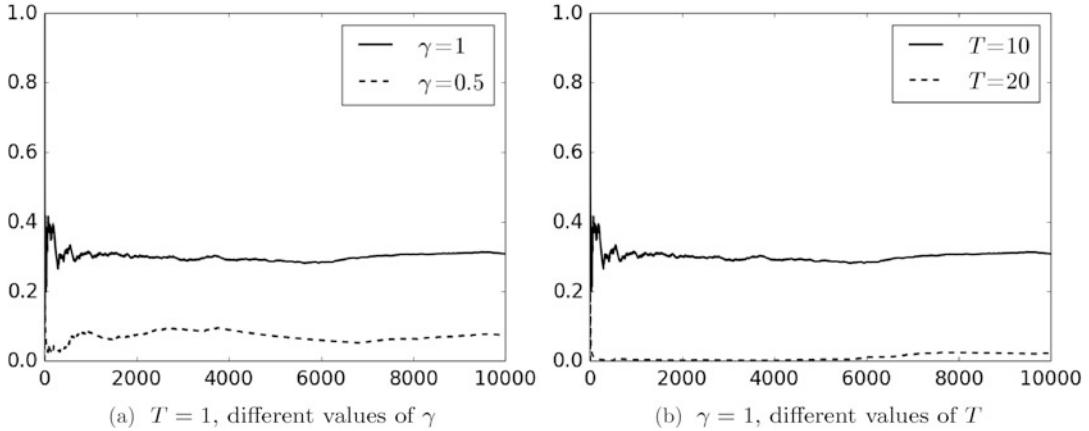


Fig. 7.3: Accept–reject probability of the independence dynamics sampler for Example 6.6 for $C_0 = 1, m_0 = 0, \epsilon = 0.08, \tau = 0.01$, and $\Gamma_0 = \gamma^2$ for different values of γ and observational time T .

plot the marginal probability distributions for $u(T)$, calculated from this converged Markov chain.

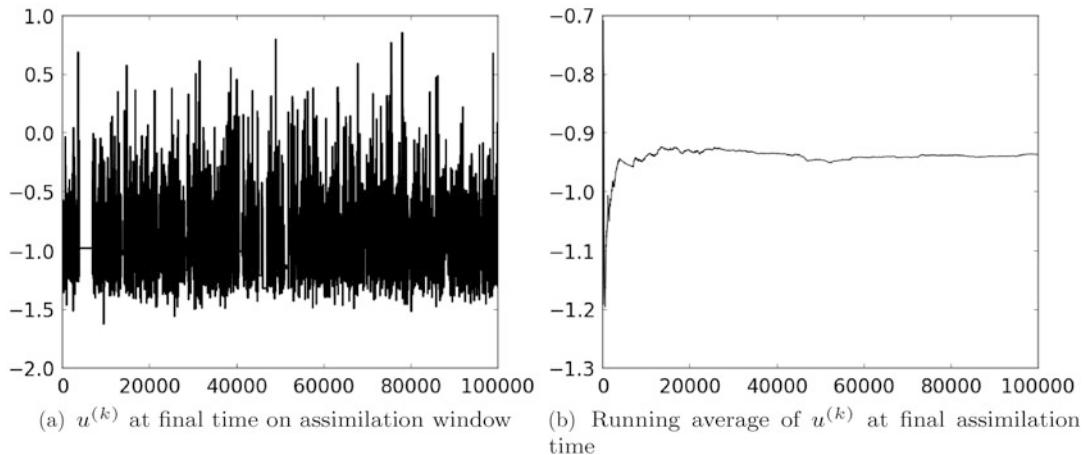


Fig. 7.4: Output and running average of the independence dynamics sampler after $K = 10^5$ steps, for Example 6.6 for $C_0 = 1, m_0 = 0, \epsilon = 0.08, \tau = 0.01$, and $\Gamma_0 = \gamma^2 = 0.5^2$ and $T = 10$.

In order to get faster convergence when we sample the posterior distribution, we turn to application of the pCN method. Unlike the independence dynamics sampler, this contains a tunable parameter that can vary the size of the proposals. In particular, the possibility of making small moves, with resultant higher acceptance probability, makes this a more flexible method than the independence dynamics sampler. In Figure 7.6, we show application of the pCN sampler, again considering Example 6.6 for $C_0 = 0, m_0 = 1, \epsilon = 0.08, \tau = 0.01$, and $\Gamma_0 = \gamma^2 = 0.5^2$ and $T = 10$, the same parameters used in Figure 7.4, while the value of β used is 0.05.

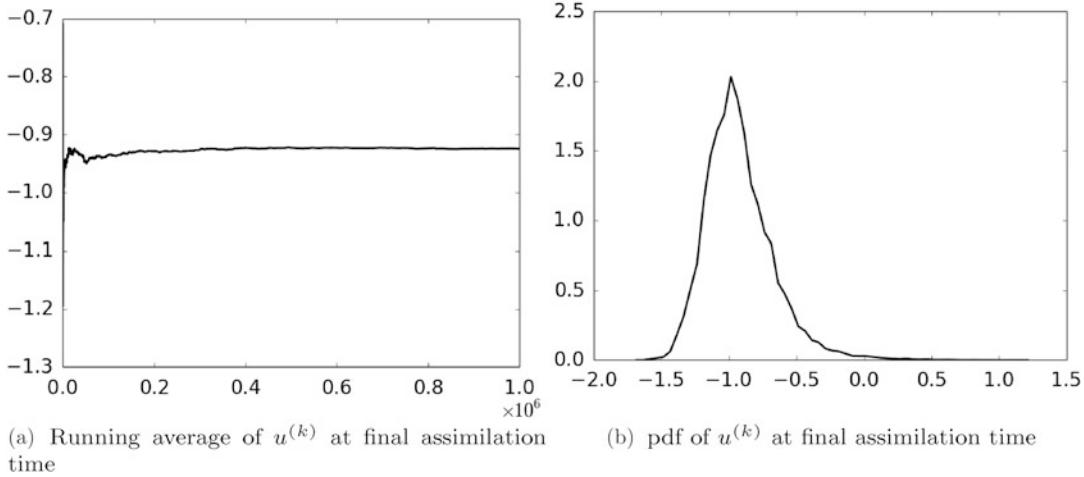


Fig. 7.5: Running average and probability density of the first and last elements of the independence dynamics sampler after $K = 10^6$ steps, for Example 6.6 for $C_0 = 0, m_0 = 1, \epsilon = 0.08, \tau = 0.01$, and $\Gamma_0 = \gamma^2 = 0.5^2$ and $T = 10$.

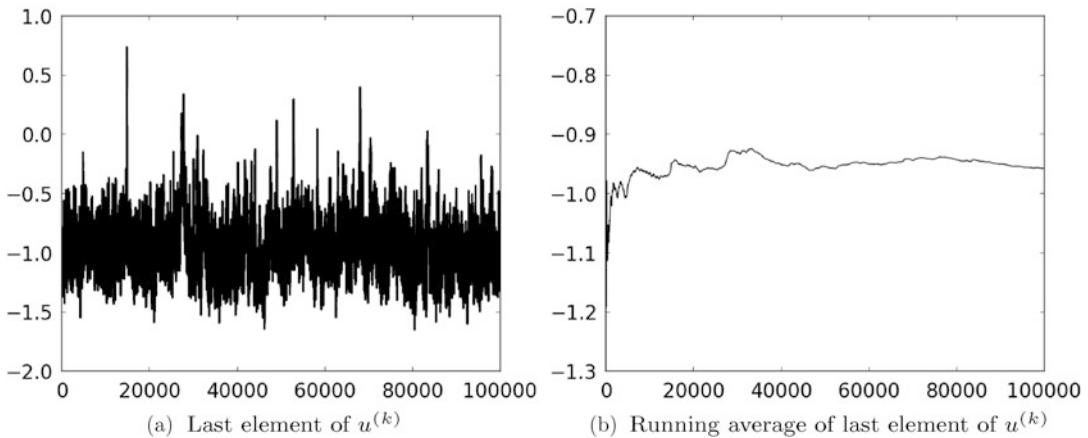


Fig. 7.6: Output and running average for the pCN sampler ($\beta = 0.05$) after $K = 10^5$ steps, for Example 6.6 for $C_0 = 0, m_0 = 1, \epsilon = 0.08, \tau = 0.01$, and $\Gamma_0 = \gamma^2 = 0.5^2$ and $T = 10$.

7.5 Bibliographic Notes

- Section 7.1 is devoted to the Kalman–Bucy filter, which first appeared in the published literature in the paper [80]. A characterization of the measure μ from Theorem 7.1 in terms of the solution of boundary value problems may be found in [64]. In particular, the paper [64] shows an explicit calculation linking the calculation as a boundary value problem to the formulas in [80]; this link corresponds to a continuous analogue of LU factorization.
- Section 7.2 concerns solution of the smoothing problem for continuous-time data assimilation by means of MCMC methods. This subject is considered in some detail in the article [63]. The reader is also pointed to the article [35] for an overview of MCMC methods for infinite-dimensional problems. In the case of deterministic dynamics, the approach

we describe corresponds to adopting a Bayesian approach to parameter estimation in SDEs; the subject of parameter estimation for SDEs is explored in the recent collection of articles [83].

- Section 7.3 concerns variational methods. In the case of stochastic dynamics, these are optimization problems over infinite-dimensional spaces, and the full power of the calculus of variations [40] and associated computational tools [112] is relevant to the development of efficient algorithms. The computational tools from [112] are also, of course, relevant in the case of deterministic dynamics, where the optimization is over a finite-dimensional space. Derivation of the appropriate cost function that should be minimized in the infinite-dimensional setting in order to define a MAP estimator is the subject of several papers by Zeitouni and coworkers in the 1980s; see [150, 43] and the references therein. The argument concerning the behavior of the objective functional in this case, for large T , underpins the proofs of posterior consistency for maximum-likelihood-based parameter estimation in SDEs [85]. For further discussion of posterior consistency, see [43].

7.6 Exercises

1. Find the exact solution of the equation

$$\frac{dv}{dt} = v(1 - v^2)$$

from Example 6.6 when $\epsilon = 0$. Using this exact solution, modify the code used in `p3c.m`, which uses a numerical approximation of the equation (6.22), to produce an improved approximation of the posterior distribution shown in Figure 6.9a with $T = 10$.

2. Use program `p5c.m` to study the acceptance probability of the independence dynamics sampler, for fixed $\epsilon > 0$, as the noise level in the observation equation is increased. Report your findings and offer an explanation.
3. Use program `p6c.m` to study the acceptance probability of the independence dynamics sampler, for fixed $\epsilon > 0$, as the proposal variance β is varied in $[0, 1]$. Report your findings and offer an explanation.
4. Define a pCN dynamics sampler for the posterior distribution of Theorem 6.13, analogously to what is done in discrete time in Section 3.2.4. Specify the proposal and the form of the acceptance probability.
5. Write a program to sample from the posterior distribution of Theorem 6.13, using the pCN dynamics sampler from the previous exercise. Apply the method to the same example considered in program `p6c.m`. Compare the output in the form of time traces, as well as time-average traces, as in Figure 7.6.

Chapter 8

Continuous Time: Filtering Algorithms

In this chapter, we describe various algorithms for determination of the filtering distribution μ^t in continuous time. We begin in Section 8.1 with the Kalman–Bucy filter, which provides an exact algorithm for linear problems. Since the filtering distribution is Gaussian in this case, the distribution is entirely characterized by the mean and covariance; the algorithm comprises a system of differential equations for the mean and the covariance. In Section 8.2, we discuss the approximate Gaussian methods introduced in Section 4.2 in the discrete-time setting. Similarly to the case of the Kalman–Bucy filter, we again obtain a differential equation for the mean; for the extended Kalman (ExKF) filter, we also obtain an equation for the covariance, and for the ensemble Kalman filter (EnKF), we have a system of differential equations coupled through their empirical mean and covariance. In Section 8.3, we discuss how the particle filter methodology introduced in Section 4.3 extends to the continuous case, while in Section 8.4, we study the long-time behavior of some of the filtering algorithms discussed in the previous sections. Finally, in Section 8.5, we present some numerical illustrations and conclude with bibliographic notes and exercises in Sections 8.6 and 8.7 respectively.

It is helpful to recall the form of the continuous-time data-assimilation problem. The signal is governed by the SDE from (6.4):

$$\begin{aligned}\frac{dv}{dt} &= f(v) + \sqrt{\Sigma_0} \frac{dB_v}{dt}, \\ v(0) &\sim N(m_0, C_0);\end{aligned}$$

the data is generated by the SDE (6.5):

$$\begin{aligned}\frac{dz}{dt} &= h(v) + \sqrt{\Gamma_0} \frac{dB_z}{dt}, \\ z(0) &= 0.\end{aligned}$$

We let z^t denote $\{z(s)\}_{s \in [0,t]}$, the data accumulated up to time t , and we are interested in the probability measure μ^t governing $v(t)|z^t$, and in particular, in updating this measure sequentially in time. This is the filtering problem. In principle, the Kushner–Stratonovich and Zakai equations provide the solution to the filtering problem, but in general, they do not have closed-form solutions. Thus algorithms are required to approximate their solution. The filtering algorithms that we describe in the remainder of the section attempt to do this.

Electronic supplementary material The online version of this chapter (doi: [10.1007/978-3-319-20325-6_8](https://doi.org/10.1007/978-3-319-20325-6_8)) contains supplementary material, which is available to authorized users.

We highlight here the fact that the Kushner–Stratonovich and Zakai equations are stochastic PDEs in spatial dimension n , the size of the state space. Thus their solution poses formidable challenges for high-dimensional problems.

8.1 Linear Gaussian Problems: The Kalman–Bucy Filter

Although the Kushner–Stratonovich and Zakai equations do not, in general, have closed-form solutions, they do have such solutions for linear models; this is entirely analogous to the discrete-time setting and stems from the fact that the desired filtering distribution is Gaussian. The resulting algorithm for the mean and covariance is the Kalman–Bucy filter, which we now describe.

Consider equations (6.4) and (6.5), where $f(v) = Lv$, $h(v) = Hv$, for $L \in \mathbb{R}^{n \times n}$, $H \in \mathbb{R}^{m \times n}$ of full rank m . Then (6.4) and (6.5) become

$$\frac{dv}{dt} = Lv + \sqrt{\Sigma_0} \frac{dB_v}{dt}, \quad v(0) \sim N(m_0, C_0), \quad (8.3a)$$

$$\frac{dz}{dt} = Hv + \sqrt{\Gamma_0} \frac{dB_w}{dt}, \quad z(0) = 0. \quad (8.3b)$$

Theorem 8.1. *The filtering distribution μ^t for $v(t)|z^t$ governed by (8.3) is Gaussian with mean m and covariance C solving the Kalman–Bucy filter*

$$\frac{dm}{dt} = Lm + CH^T \Gamma_0^{-1} \left(\frac{dz}{dt} - Hm \right), \quad m(0) = m_0,$$

$$\frac{dC}{dt} = LC + CL^T + \Sigma_0 - CH^T \Gamma_0^{-1} HC, \quad C(0) = C_0.$$

Sketch Proof We give references to the rigorous derivation of Kalman–Bucy filtering in the bibliographic notes at the end of the chapter. Here we derive the filter by a formal discretization argument, with no proof, since this provides a useful way to understand the structure of the filter. In particular, we consider the discrete-time Kalman filter of Section 4.1, and the prediction and analysis steps given by (4.4), (4.5) and (4.7), (4.8) respectively, with $M = I + \tau L$ and the other scalings detailed in (6.1). The prediction steps (4.4) and (4.5) give, to leading order in τ ,

$$\hat{m}_{j+1} = m_j + \tau Lm_j, \quad (8.4)$$

$$\begin{aligned} \hat{C}_{j+1} &= (I + \tau L)C_j(I + \tau L)^T + \tau \Sigma_0 \\ &= C_j + \tau(LC_j + C_j L^T + \Sigma_0) + \mathcal{O}(\tau^2). \end{aligned} \quad (8.5)$$

Now using the analysis step from Corollary 4.2, again to leading order and substituting (8.4) and (8.5), we obtain

$$\begin{aligned} d_{j+1} &= (y_{j+1} - Hm_j) + \mathcal{O}(\tau), \\ S_{j+1} &= \frac{1}{\tau} \Gamma_0^{-1} + \mathcal{O}(1), \\ K_{j+1} &= \tau C_j H^T \Gamma_0^{-1} + \mathcal{O}(\tau^2), \\ m_{j+1} &= m_j + \tau Lm_j + \tau C_j H^T \Gamma_0^{-1} (y_{j+1} - Hm_j), \\ C_{j+1} &= C_j + \tau(LC_j + C_j L^T + \Sigma_0) - \tau C_j H^T \Gamma_0^{-1} HC_j + \mathcal{O}(\tau^2). \end{aligned}$$

Recalling that $y_{j+1} = \tau^{-1}(z_{j+1} - z_j)$ and passing to the limit gives

$$\begin{aligned}\frac{dm}{dt} &= Lm + CH^T \Gamma_0 \left(\frac{dz}{dt} - Hm \right), \\ \frac{dC}{dt} &= LC + CL^T + \Sigma_0 - CH^T \Gamma_0^{-1} HC,\end{aligned}$$

which concludes the proof sketch. \square

8.2 Approximate Gaussian Filters

Here we discuss the family of approximate Gaussian filters introduced in Section 4.2, generalizing to continuous time. Our aim is to ascertain the form of continuous-time limits under the scalings detailed in (6.1). Our aim is not to prove theorems about the limiting process, but rather to give an understanding of what the natural continuous-time limiting processes are. We thus use the environment “result” rather than “theorem” to highlight the fact that the forms of the continuous-time limits are derived only formally and not, currently, rigorously proved in the published literature; however, it would not be difficult to make rigorous proofs based on these results.

The starting point of our investigations is equation (4.13), which is based on the assumption that $\mathbb{P}(u_{j+1}|Y_j) = N(\Psi(m_j), \widehat{C}_{j+1})$. In addition to the expression for the update of m_j (4.13a) using Bayes’s rule one sees that the new covariance C_{j+1} satisfies

$$C_{j+1} = (I - K_{j+1}H)\widehat{C}_{j+1}. \quad (8.6)$$

We will now proceed in a similar fashion as in the case of the Kalman–Bucy filter: we derive a differential equation for the time evolution of the mean and covariance of the different approximate Gaussian filters studied in Section 4.2. The resulting Gaussian measures should be viewed as attempts to approximate the true filtering distribution μ^t .

8.2.1. 3DVAR

Result 8.2. Consider the filtering distribution for the 3DVAR algorithm (4.14) arising in the case of deterministic dynamics ($\Sigma = 0$) and linear observations ($h(v) = Hv$). Under the scalings detailed in (6.1), and in the continuous-time limit ($\tau \rightarrow 0$), the corresponding limiting filtering distribution is Gaussian with mean m and covariance C satisfying

$$\frac{dm}{dt} = f(m) + CH^T \Gamma_0^{-1} \left(\frac{dz}{dt} - Hm \right), \quad m(0) = m_0, \quad (8.7a)$$

$$\frac{dC}{dt} = 0, \quad C(0) = \widehat{C}. \quad (8.7b)$$

Derivation We begin our derivation by observing that (4.13) implies that

$$m_{j+1} = \Psi(m_j) + \widehat{C}_{j+1} H^T (\Gamma + H\widehat{C}_{j+1} H^T)^{-1} (y_{j+1} - H\Psi(m_j)).$$

We now apply the scalings appropriate for a continuous-time limit and in particular set

$$\Psi(m) = m + \tau f(m) + \mathcal{O}(\tau^2).$$

In addition, for the case of 3DVAR, we have $\widehat{C}_{j+1} = \widehat{C}$ for all j , which implies that

$$\widehat{C}_{j+1} - \widehat{C}_j = 0.$$

Finally, using the scalings from equation (6.1) and recalling that $y_{j+1} = \tau^{-1}(z_{j+1} - z_j)$, we have that

$$\begin{aligned} \frac{m_{j+1} - m_j}{\tau} &= f(m_j) + \widehat{C}_{j+1} H^T (\Gamma_0 + \tau H \widehat{C}_{j+1} H^T)^{-1} \left(\frac{z_{j+1} - z_j}{\tau} - H m_j \right) + \mathcal{O}(\tau), \\ \frac{\widehat{C}_{j+1} - \widehat{C}_j}{\tau} &= 0. \end{aligned}$$

By taking the limit $\tau \rightarrow 0$, we obtain

$$\begin{aligned} \frac{dm}{dt} &= f(m) + CH^T \Gamma_0^{-1} \left(\frac{dz}{dt} - Hm \right), \quad m(0) = m_0, \\ \frac{dC}{dt} &= 0, \quad C(0) = \widehat{C}, \end{aligned}$$

where we have identified C_0 with \widehat{C} . This completes our derivation. \square

8.2.2. Extended Kalman Filter

The 3DVAR algorithm imposes a fixed covariance on the model in the prediction step of the algorithm, implying also a fixed covariance in the analysis step. The extended Kalman filter attempts to improve on this by propagating the covariance in the prediction step according to the linearized dynamics. In the continuous-time limit, we obtain the following.

Result 8.3. *Consider the filtering distribution for the extended Kalman Filter from Section 4.2.2 in the case of stochastic dynamics and linear observations ($h(v) = Hv$). Under the scalings detailed in (6.1) and in the continuous-time limit ($\tau \rightarrow 0$), the corresponding limiting filtering distribution is Gaussian with mean m and covariance C satisfying*

$$\begin{aligned} \frac{dm}{dt} &= f(m) + CH^T \Gamma_0^{-1} \left(\frac{dz}{dt} - Hm \right), \quad m(0) = m_0, \\ \frac{dC}{dt} &= Df(m)C + C(Df(m))^T + \Sigma_0 - CH^T \Gamma_0^{-1} HC, \quad C(0) = C_0. \end{aligned}$$

Derivation From the formulas given in Section 4.2.2, we have

$$\widehat{C}_{j+1} = D\Psi(m_j)C_jD\Psi(m_j)^T + \Sigma,$$

and using (4.13) and (8.6),

$$\begin{aligned} m_{j+1} &= \Psi(m_j) + \widehat{C}_{j+1} H^T (\Gamma + H \widehat{C}_{j+1} H^T)^{-1} (y_{j+1} - H \Psi(m_j)), \\ \widehat{C}_{j+1} &= D\Psi(m_j) (I - K_j H) \widehat{C}_j D\Psi(m_j)^T + \Sigma. \end{aligned}$$

To deduce the continuous-time limit, we set $\Psi(m) = m + \tau f(m) + \mathcal{O}(\tau^2)$ and impose (6.1). This yields

$$\begin{aligned} D\Psi(m_j) &= I + \tau Df(m_j) + \mathcal{O}(\tau^2), \\ (I - K_j H) &= I - \tau \widehat{C}_j H^T (\Gamma_0 + \tau H \widehat{C}_j H^T)^{-1} H + \mathcal{O}(\tau^2). \end{aligned}$$

Combining the two previous sets of equations and recalling that $y_{n+1} = \tau^{-1}(z_{n+1} - z_n)$, we obtain

$$\begin{aligned} \frac{m_{j+1} - m_j}{\tau} &= f(m_j) + \widehat{C}_{j+1} H^T (\Gamma_0 + \tau H \widehat{C}_{j+1} H^T)^{-1} \left(\frac{z_{j+1} - z_j}{\tau} - H m_j \right) + \mathcal{O}(\tau), \\ \frac{\widehat{C}_{j+1} - \widehat{C}_j}{\tau} &= Df(m_j) \widehat{C}_j + \widehat{C}_j Df(m_j)^T - \widehat{C}_j H^T (\Gamma_0 + \tau H \widehat{C}_j H^T)^{-1} H \widehat{C}_j + \Sigma_0 + \mathcal{O}(\tau). \end{aligned}$$

By taking the limit $\tau \rightarrow 0$, we obtain

$$\begin{aligned} \frac{dm}{dt} &= f(m) + CH^T \Gamma_0^{-1} \left(\frac{dz}{dt} - Hm \right), \quad m(0) = m_0, \\ \frac{dC}{dt} &= Df(m)C + C(Df(m))^T + \Sigma_0 - CH^T \Gamma_0^{-1} HC, \quad C(0) = C_0, \end{aligned}$$

as required. \square

8.2.3. Ensemble Kalman Filter

As already discussed in Chapter 4, the ensemble Kalman filter differs from the extended Kalman filter and 3DVAR in that instead of using an appropriate minimization procedure to update a single estimate of the mean, a minimization principle is used to generate an ensemble of particles all of which satisfy the model/data compromise inherent in the minimization; these are coupled through the empirical covariance used to weight the minimization. Thus in studying the EnKF in continuous time, it is natural to derive an SDE for each of the particles, instead of deriving a single equation for the mean and the covariance as in Results 8.2 and 8.3. In deriving the continuous-time limit for each of the particles, it will be useful to rewrite the ensemble Kalman filter from Section 4.2.3 using the family of minimization principles $\mathbf{l}_{\text{filter},nn}$ given by (4.15), with $n = 1, \dots, N$ indexing the particles. Using such an interpretation leads to the following equation:

$$\widehat{C}_{j+1}^{-1} (I + \widehat{C}_{j+1} H^T \Gamma^{-1} H) v_{j+1}^{(n)} = \widehat{C}_{j+1}^{-1} \widehat{v}_{j+1}^{(n)} + H^T \Gamma^{-1} y_{j+1}^{(n)},$$

with \widehat{C}_{j+1} the predictive covariance found from the properties of the predictive ensemble; in this derivation we simply assume that \widehat{C}_{j+1} is invertible and return to address this issue after the derivation. Applying \widehat{C}_{j+1} allows us to rewrite the analysis step of the ensemble Kalman Filter in the following form:

$$(I + \widehat{C}_{j+1} H^T \Gamma^{-1} H) v_{j+1}^{(n)} = \widehat{v}_{j+1}^{(n)} + \widehat{C}_{j+1} H^T \Gamma^{-1} y_{j+1}^{(n)}, \quad (8.8a)$$

$$y_{j+1}^{(n)} = y_{j+1} + \eta_{j+1}^{(n)}, \quad n = 1, \dots, N, \quad (8.8b)$$

where $\widehat{v}_{j+1}^{(n)}$, \widehat{m}_{j+1} , and \widehat{C}_{j+1} are given by the prediction step detailed in Section 4.2.3. We now have the following result:

Result 8.4. Consider the ensemble Kalman Filter from Section 4.2.3 in the case of stochastic dynamics and linear observations ($h(v) = Hv$). Under the scalings detailed in (6.1) and in the continuous-time limit ($\tau \rightarrow 0$), the particles evolve according to the following set of coupled SDEs, for $n = 1, \dots, N$:

$$\frac{dv^{(n)}}{dt} = f(v^{(n)}) + C(v) H^T \Gamma_0^{-1} \left(\frac{dz^{(n)}}{dt} - Hv^{(n)} \right) + \Sigma_0^{1/2} dB_v, \quad (8.9a)$$

$$\frac{dz^{(n)}}{dt} = Hv + \Gamma_0^{1/2} \left(\frac{dW^{(n)}}{dt} + \frac{dB_z}{dt} \right), \quad (8.9b)$$

where $W^{(1)}, \dots, W^{(N)}, B_z, B_u$ are mutually independent standard Wiener processes. The mean $m(v)$ and covariance $C(v)$ are defined empirically from the particles $v = \{v^{(n)}\}_{n=1}^N$ as follows:

$$m(v) = \frac{1}{N} \sum_{n=1}^N v^{(n)}, \quad (8.10a)$$

$$C(v) = \frac{1}{N-1} \sum_{n=1}^N (v^{(n)} - m)(v^{(n)} - m)^T. \quad (8.10b)$$

Derivation We begin our derivation by noting the formulation of the analysis step given in (8.8) and employing the definition of the particle prediction step to give

$$\begin{aligned} v_{j+1}^{(n)} - v_j^{(n)} &= \widehat{v}_{j+1}^{(n)} - v_j^{(n)} - \widehat{C}_{j+1} H^T \Gamma^{-1} H v_{j+1}^{(n)} + \widehat{C}_{j+1} H^T \Gamma^{-1} y_{j+1}^{(n)} \\ &= \Psi(v_{j+1}^{(n)}) - v_j^{(n)} - \widehat{C}_{j+1} H^T \Gamma^{-1} H v_{j+1}^{(n)} + \widehat{C}_{j+1} H^T \Gamma^{-1} y_{j+1}^{(n)} + \xi_j^{(n)}. \end{aligned}$$

Now using the scalings from equation (6.1), we have that

$$\Psi(v) = v + \tau f(v) + \mathcal{O}(\tau^2), \quad \xi_j^{(n)} = \sqrt{\tau} \Sigma_0^{1/2} \widehat{\xi}_{j+1}^{(n)},$$

and thus we obtain

$$v_{j+1}^{(n)} - v_j^{(n)} = \tau(f(v_j^{(n)}) - \widehat{C}_{j+1} H^T \Gamma_0^{-1} H v_{j+1}^{(n)}) + \tau \widehat{C}_{j+1} H^T \Gamma_0^{-1} y_{j+1}^{(n)} + \sqrt{\tau} \Sigma_0^{1/2} \widehat{\xi}_{j+1}^{(n)} + \mathcal{O}(\tau^2),$$

where $\widehat{\xi}_j^{(n)}$ is $N(0, I)$ distributed. Now using the rescaling $y_{j+1}^{(n)} = \tau^{-1}(z_{j+1}^{(n)} - z_j^{(n)})$, we have the coupled difference equations

$$v_{j+1}^{(n)} - v_j^{(n)} = \tau(f(v_j^{(n)}) - \widehat{C}_{j+1} H^T \Gamma_0^{-1} H v_{j+1}^{(n)}) + \widehat{C}_{j+1} H^T \Gamma_0^{-1} (z_{j+1}^{(n)} - z_j^{(n)}) + \sqrt{\tau} \Sigma_0^{1/2} \widehat{\xi}_{j+1}^{(n)}, \quad (8.11)$$

$$z_{j+1}^{(n)} - z_j^{(n)} = \tau H v_{j+1} + \sqrt{\tau} \Gamma_0^{1/2} (\widehat{\eta}_{j+1}^{(n)} + \widehat{\eta}_{j+1}), \quad (8.12)$$

where $\widehat{\xi}_{j+1}^{(n)}$, $\widehat{\eta}_{j+1}^{(n)}$, and $\widehat{\xi}_{j+1}$ are independent $N(0, I)$ -distributed random variables. In addition, we have

$$\widehat{m}_{j+1} = \frac{1}{N} \sum_{n=1}^N v_{j+1}^{(n)} + \mathcal{O}(\tau),$$

so in the limit of $\tau \rightarrow 0$, we have that $(\widehat{m}_{j+1}, \widehat{C}_{j+1}) \rightarrow (m, C)$ given by equation (8.10). Furthermore, we see that the coupled difference equations given by (8.11) form a mixed implicit-explicit Euler–Maruyama-type scheme for the system of SDEs (8.9), and thus in the limit of $\tau \rightarrow 0$, we obtain the desired equations. \square

Recall that in the preceding derivation, we made the assumption that \widehat{C}_{j+1} is invertible. However, this might not be the case; indeed, it cannot be the case if the dimension of the state space exceeds the number of particles. However, one can still obtain the key equations (8.8) in this case, by applying the following lemma to the analysis formula in Section 4.2.3.

Lemma 8.5. *Assume that Γ is invertible. Then*

$$(I - K_{j+1}H)^{-1} = (I + \widehat{C}_{j+1}H^T\Gamma^{-1}H)$$

and

$$(I - K_{j+1}H)^{-1}K_{j+1} = \widehat{C}_{j+1}H^T\Gamma^{-1}.$$

Proof We begin the proof by noting that using the Woodbury matrix identity from Lemma 4.4, we have, for

$$S_{j+1} := (\Gamma + H\widehat{C}_{j+1}H^T),$$

that

$$S_{j+1}^{-1} = \Gamma^{-1} - \Gamma^{-1}H(\widehat{C}_{j+1}^{-1} + H^T\Gamma^{-1}H)^{-1}H^T\Gamma^{-1},$$

where we note that S_{j+1} is invertible because Γ is. Assume that C_{j+1} is invertible; we will relax this assumption below. Thus $Z := I - K_{j+1}H = I - \widehat{C}_{j+1}H^TS_{j+1}^{-1}H$ can be written as

$$\begin{aligned} Z &= I - \widehat{C}_{j+1}H^T\Gamma^{-1}H - \widehat{C}_{j+1}H^T\Gamma^{-1}H(\widehat{C}_{j+1}^{-1} + H^T\Gamma^{-1}H)^{-1}H^T\Gamma^{-1}H \\ &= I - \widehat{C}_{j+1}B - \widehat{C}_{j+1}B(\widehat{C}_{j+1}^{-1} + B)^{-1}B \\ &= I - \widehat{C}_{j+1}B(I - (\widehat{C}_{j+1}^{-1} + B)^{-1}B) \\ &= I - \widehat{C}_{j+1}B(\widehat{C}_{j+1}^{-1} + B)^{-1}\widehat{C}_{j+1}^{-1}, \end{aligned}$$

where $B = H^T\Gamma^{-1}H$. Manipulating this expression further, we see that

$$Z(I + \widehat{C}_{j+1}B) = I.$$

This identity may be derived even if C_{j+1} is not invertible simply by adding ϵI to C_{j+1} in the preceding derivation and then letting $\epsilon \rightarrow 0$. The preceding identity implies that

$$(I - K_{j+1}H)^{-1} = (I + \widehat{C}_{j+1}B) = (I + \widehat{C}_{j+1}H^T\Gamma^{-1}H),$$

which concludes the proof for our first equation. Now using this equation, it is easy to see that

$$K_{j+1} = H^{-1} - (I + \widehat{C}_{j+1}H^T\Gamma^{-1}H)^{-1}H^{-1}$$

and thus

$$(I - K_{j+1}H)K_{j+1} = (I + \widehat{C}_{j+1}H^T\Gamma^{-1}H)H^{-1} - H^{-1} = \widehat{C}_{j+1}H^T\Gamma^{-1},$$

which concludes the proof for our second equation. \square

8.3 The Particle Filter

The particle filter in continuous time faces many of the same issues arising in discrete time, as outlined in Section 4.3. In the basic version of the method, analogous to the bootstrap filter of Section 4.3.2, the particles evolve in continuous time according to the SDE (6.4). The particles are weighted according to (6.31), which reflects the change of measure required to take the solution of the SDE into the solution of the SDE conditioned on the observations given by (6.5). As in discrete time, it is helpful to resample from the resulting distribution in order to obtain an approximation with significant weight near the data. References to detailed literature on the subject are given in Section 8.6.

8.4 Large-Time Behavior of Filters

Here we provide some simple examples that illustrate issues relating to the large-time behavior of filters. The discussion from the preamble of Section 4.4 also applies here in continuous time. In particular, the approximate Gaussian filters do not perform well as measured by the Bayesian quality assessment test of Section 2.7 but may perform well as measured by the signal estimation quality assessment test. Also, similarly to the situation in discrete time, the Kalman–Bucy filter for linear problems and the particle filter give accurate approximations of the true posterior distribution, in the latter case in the large-particle limit. The purpose of this section is to illustrate these issues.

8.4.1. The Kalman–Bucy Filter in One Dimension

We consider the case of one-dimensional deterministic linear dynamics (8.3) with $\Sigma_0 = 0$ and

$$f(v) = \ell v, \quad h(v) = v,$$

while we will also assume that

$$\Gamma_0 = \gamma^2, C_0 = c_0.$$

Thus the filter aims to reconstruct signal $v(t)$ solving the equation

$$\frac{dv}{dt} = \ell v$$

from knowledge of $\{z(s)\}_{0 \leq s \leq t}$, where

$$z(s) = \int_0^s v(\tau)d\tau + \gamma B_w(s).$$

Even though the variance of $B_w(t)$ grows linearly with t , we will show that the variance of the Kalman–Bucy filter is asymptotically zero, for $\ell \leq 0$, or of order $\mathcal{O}(\gamma^2)$ otherwise.

With these definitions, the Kalman–Bucy filter of Theorem 8.1 becomes

$$\begin{aligned}\frac{dm}{dt} &= \ell m + \gamma^{-2} c \left(\frac{dz}{dt} - m \right), \quad m(0) = m_0, \\ \frac{dc}{dt} &= 2\ell c - \gamma^{-2} c^2, \quad c(0) = c_0.\end{aligned}$$

Here m denotes the mean of the filter, and c , the variance.

We notice that the equation for the variance evolves independently of that for the mean, and independently of the data. Furthermore, if we define the precision p to be the inverse of c , then straightforward calculation reveals that p solves the linear equation

$$\frac{dp}{dt} = -2\ell p + \gamma^{-2}.$$

For $\ell \neq 0$, this has exact solution

$$p(t) = \exp(-2\ell t) \frac{1}{c_0} + \left(1 - \exp(-2\ell t)\right) \frac{1}{2\ell\gamma^2},$$

while for $\ell = 0$, we see that

$$p(t) = \frac{1}{c_0} + \frac{t}{\gamma^2}.$$

Thus for $\ell \leq 0$, we have $p(t) \rightarrow \infty$ as $t \rightarrow \infty$. and the asymptotic variance is zero, while for $\ell > 0$, the asymptotic variance is $2\ell\gamma^2$. In particular, if the observational variance γ^2 is small, then the asymptotic variance of the Kalman filter is $\mathcal{O}(\gamma^2)$, even when $\ell > 0$, so that the dynamics is unstable. The key point to observe, then, is that asymptotic uncertainty is small, independently of whether the underlying deterministic dynamics is stable. In words, observation can stabilize uncertainty growth in unstable systems. We study the behavior of the error in the mean in the exercises at the end of this chapter.

8.4.2. The 3DVAR Filter

Recall the 3DVAR continuous filtering algorithm (8.7). We will study the behavior of this algorithm with data $z := \{z^\dagger(t)\}_{t \in [0, \infty)}$ constructed as follows. We let $\{v^\dagger(t)\}_{t \in [0, \infty)}$ denote the exact solution of the equations (6.4) in the case $\Sigma_0 = 0$:

$$\frac{dv^\dagger}{dt} = f(v^\dagger), \quad v^\dagger(0) = v_0^\dagger. \tag{8.13}$$

We assume that the data z^\dagger is a single realization of the SDE (6.4) with $v = v^\dagger$ and in the case $h(\cdot) = H$:

$$\frac{dz^\dagger}{dt} = Hv^\dagger + \sqrt{T_0} \frac{dB_z}{dt}, \quad z^\dagger(0) = 0. \tag{8.14}$$

In order to study the continuous-time 3DVAR filter, we eliminate $z = z^\dagger$ in equation (8.7) using (8.14) to obtain

$$\frac{dm}{dt} = f(m) + CH^T \Gamma_0^{-1} H(v^\dagger - m) + CH^T \Gamma_0^{-\frac{1}{2}} \frac{dB_z}{dt}. \quad (8.15)$$

In the following theorem, expectation is with respect to the Brownian motion B_z entering the data equation (8.14).

Theorem 8.6. *Let m solve equation (8.15), let v^\dagger solve equation (8.13), assume that f is globally Lipschitz with constant L and that there exist $\lambda > 0$ and $\epsilon > 0$ such that*

$$\begin{aligned} \langle CH^T \Gamma_0^{-1} H a, a \rangle &\geq (L + \frac{1}{2}\lambda)|a|^2, \forall a \in \mathbb{R}^n, \\ \text{Tr}(\Gamma_0^{-\frac{1}{2}} H C^2 H^T \Gamma_0^{-\frac{1}{2}}) &\leq \epsilon^2. \end{aligned}$$

Then the error in the 3DVAR filter satisfies

$$\mathbb{E}|m(t) - v(t)|^2 \leq e^{-\lambda t}|m(0) - v(0)|^2 + \frac{\epsilon^2}{\lambda}(1 - e^{-\lambda t}). \quad (8.16)$$

Thus

$$\limsup_{t \rightarrow \infty} \mathbb{E}|m(t) - v(t)|^2 \leq \frac{\epsilon^2}{\lambda}. \quad (8.17)$$

Proof Define $\delta = m - v^\dagger$ and subtract equation (8.13) from equation (8.15) and apply the Itô formula to $|\delta|^2$ to obtain

$$\frac{1}{2}d|\delta|^2 + \langle CH^T \Gamma_0^{-1} H \delta, \delta \rangle dt \leq \langle f(m) - f(v^\dagger), \delta \rangle + \langle \delta, CH^T \Gamma_0^{-\frac{1}{2}} dB_z \rangle + \frac{1}{2}\text{Tr}\left(\Gamma_0^{-\frac{1}{2}} H C^2 H^T \Gamma_0^{-\frac{1}{2}}\right) dt. \quad (8.18)$$

Using the Lipschitz property of f and the definition of λ and ϵ , and taking expectations, gives

$$\frac{d\mathbb{E}|\delta|^2}{dt} \leq -\lambda\mathbb{E}|\delta|^2 + \epsilon^2. \quad (8.19)$$

Use of the Gronwall inequality gives the desired result. \square

It is interesting to consider the asymptotic behavior of this 3DVAR filter in the linear Gaussian case from the preceding subsection. We thus assume that

$$f(v) = \ell v, \quad h(v) = v,$$

and that

$$\Gamma_0 = \gamma^2, C = \eta^{-1}\gamma^2 I.$$

We assume that $\gamma^2 \ll 1$ and that $\ell > 0$. Our scaling of C proportional to γ^2 is motivated by the fact that the Kalman–Bucy filter has asymptotic variance on this scale. Theorem 8.6 applies, provided η is chosen to satisfy

$$\eta \leq (\ell + \frac{1}{2}\lambda)^{-1},$$

and then (8.17) shows that the mean-square error in the filter is bounded by $\eta^{-2}\lambda^{-1}\gamma^2$. Thus the asymptotic error of the 3DVAR filter scales in the same way as the error Kalman filter (which we study in the exercises) if the covariance C is tuned appropriately.

8.5 Illustrations

In this section, the output of the various filtering algorithms will be presented. The text is minimal, since the filters and their respective behaviors relative to one another are analogous to their discrete-time counterparts as detailed in Chapter 4.

Figure 8.1 shows application of the Kalman–Bucy filter to equation (6.21), in dimension $n = 2$, with

$$\Lambda = \begin{pmatrix} -1 & 1 \\ -1 & -1 \end{pmatrix}.$$

The observation operator is $H = (1, 0)^T$, so that the second component is unobserved. Figure 8.1a shows that the unobserved component is accurately recovered in the long-time limit, despite the fact that the filter is initialized far from the truth. Figures 8.1b and 8.1c show the asymptotic behavior of the covariance and the square of the Euclidean error between the mean and the true signal underlying the data; both are shown pathwise and in running average form.

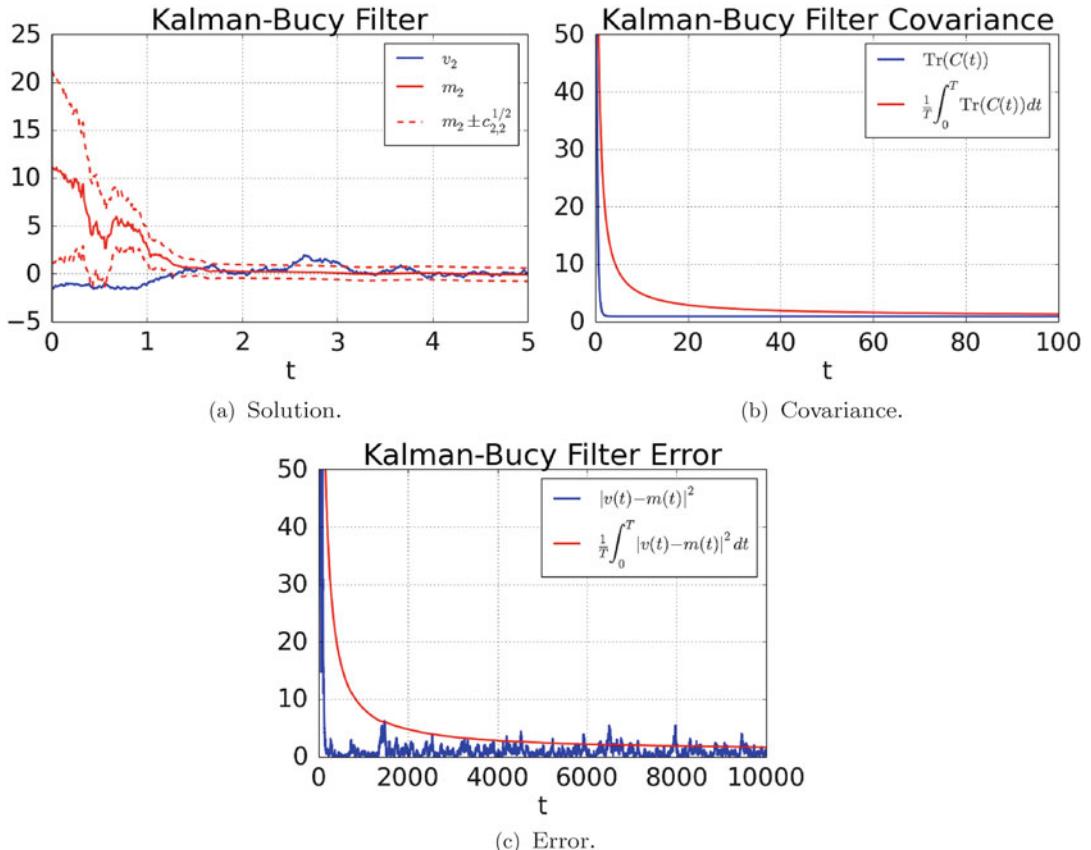


Fig. 8.1: Kalman–Bucy filter for Example 6.4 with Λ given by (8.20) with $\gamma = \sigma = 1$, as given in Section 8.20.

The next figures all concern the application of approximate Gaussian filters, together with the particle filter, to Example 6.6. Figures 8.2 and 8.3 concern application of the 3DVAR filter for $(\gamma, \sigma, \eta) = (0.3, 0.3, 0.1)$ and $(0.1, 0.3, 0.1)$ respectively, where γ^2 and σ^2 are the variance of the observations and the model dynamics respectively, and $C = \eta^{-1}\gamma^2$ is the fixed variance defining the filter. Notice the decrease in error from Figure 8.2 to 8.3 resulting from the decrease in γ . This is consistent with Theorem 8.6. Similar behavior is observed for the other filters, as shown in Figures 8.4–8.8, where the extended Kalman filter, two forms of the ensemble Kalman filter, and two forms of the particle filter are displayed. The extended Kalman filter has, arguably, the best performance; but it is a method that does not scale well to high-dimensional problems.

Remark 8.7. *It is important to remark that stability can be a significant issue, especially for filters in continuous time when complex nonlinear models are considered. As examples of this, consider the continuous-time extended Kalman filter applied to the chaotic dynamical systems Lorenz '63 (6.23) and Lorenz '96 (6.24), as presented in Section 6.2. In both cases, whether instability is observed depends on the observation operator, for example. Figures 8.9 and 8.10 show the second component (left) and mean-square error (right) of the Lorenz '63 model (6.23) with $\sigma = 2$ and $\gamma = 0.2$. In both cases, we make a scalar linear observation $h(v) = Hv$. The difference is that in Figure 8.9, the observation operator is given by $H = (1, 0, 0)$, while in Figure 8.10, the observation operator is given by $H = (0, 0, 1)$. Figures 8.11 and 8.12 show the second component (left) and mean-square error (right) of the Lorenz '96 model (6.24) with $\sigma = 1$ and $\gamma = 0.1$. The difference is that in Figure 8.11, we observe two out of every three degrees of freedom, again linearly, while in Figure 8.12, we observe one out of every three degrees of freedom, also linearly. Note that for both Lorenz models, for observations that are insufficient to keep the filter close to the truth, large excursions in the error can occur. These large excursions can easily induce numerical instabilities and destabilize the algorithms unless care is taken in the choice of integrator and time step.*



8.6 Bibliographic Notes

- In Section 8.1, we consider the continuous-time limit of the Kalman filter. This is the celebrated Kalman–Bucy filter, which was published in [80], the year after Kalman’s original paper in the discrete-time setting [79], as described in Section 4.1. The Kalman–Bucy smoother concerns the related continuous-time smoothing problem; it may be solved by a continuous-time analogue of LU factorization, in which the first triangular sweep corresponds to application of the Kalman–Bucy filter—see [64] and the discussion at the end of the previous chapter. Theorem 8.1 can also be derived from the Kushner–Stratonovich or Zakai equation equation of Theorem 6.16 by computing moments.
- Section 8.2 concerns approximate Gaussian filters and, more specifically, filters that are derived as continuous-time limits of the discrete-time approximate Gaussian filters of Section 4.2. The idea of deriving continuous-time limits of the 3DVAR, extended, and ensemble Kalman filters was developed systematically in the papers [21, 82]. Furthermore, those papers, along with the paper [88], contain analyses of the large-time stability and accuracy of the filters, with results similar in spirit to Theorem 4.10.
- Section 8.3 concerns the continuous-time particle filter. This methodology for solving continuous-time filtering problems arising in SDEs can be viewed as constituting a particle method for solution of the underlying stochastic PDE (Zakai or Kushner–Stratonovich)

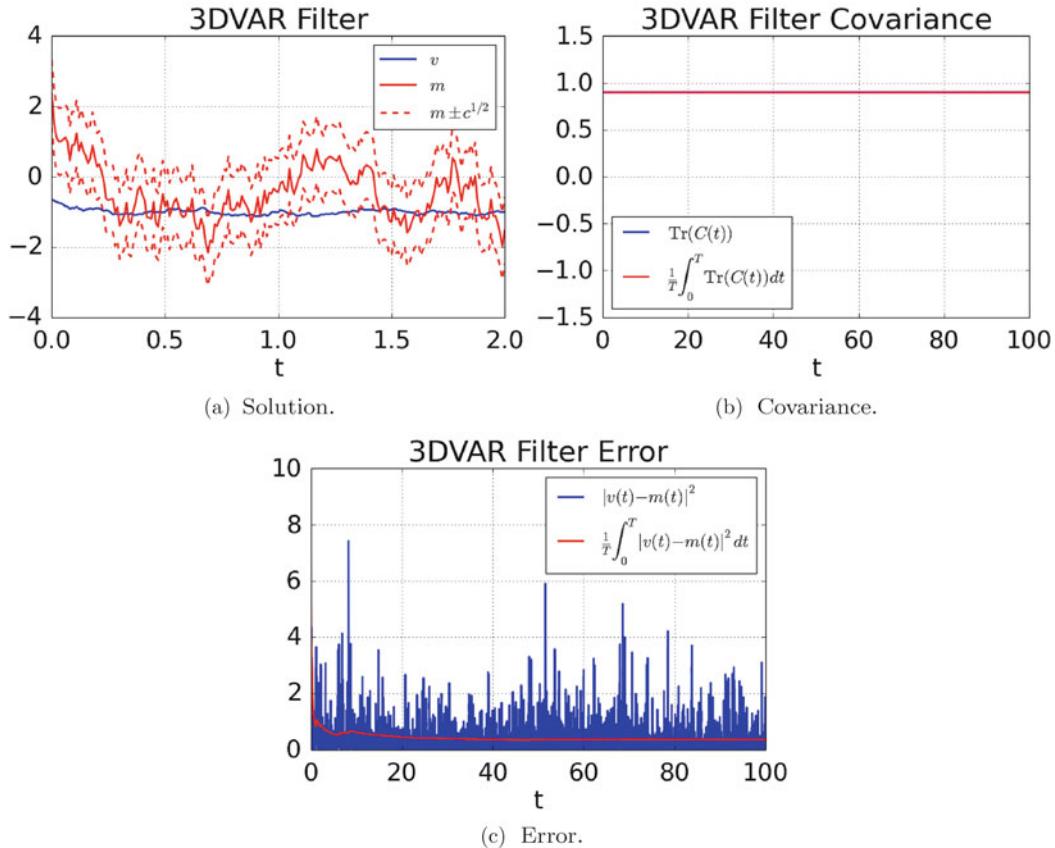


Fig. 8.2: Continuous 3DVAR filter for double-well Example 6.5 with $\gamma = \sigma = 0.3$, as given in Section 9.3.2.

introduced in Section 6.4 and governing the probability density of $v|z^t$. The method is analyzed in detail in [9] and in [45]; these two books also provide copious references to the literature on this subject.

- Section 8.4 concerns stability of filters. In Section 8.4.1, we study the one-dimensional Kalman filter, while Section 8.4.2 concerns the 3DVAR filter. The example and theorem, respectively, covered in these two subsections are entirely analogous to those in discrete time in Section 4.4.

The example from Section 8.4.1 concerning the Kalman–Bucy filter is very specific to one-dimensional deterministic dynamics. However, the general setting is thoroughly studied, as in discrete time, and the reader is directed to the book [86], concerning Riccati equations, for details. Theorem 8.6 is a simplified version of a result first proved in the context of the Navier–Stokes equation in [21], and then for the Lorenz ’63 model in [88]. The first stability analysis in continuous time concerned noise-free data and a synchronization filter in which the observed variables are simply inserted into the governing equations for the unobserved variables, giving rise to a nonautonomous dynamical system for the unobserved variables [116]; this analysis forms the backbone of the analyses of the 3DVAR filter for the Navier–Stokes and Lorenz ’63 models. The analysis was recently extended to the Lorenz ’96 model in [87]. The synchronization filter is discussed in discrete time in

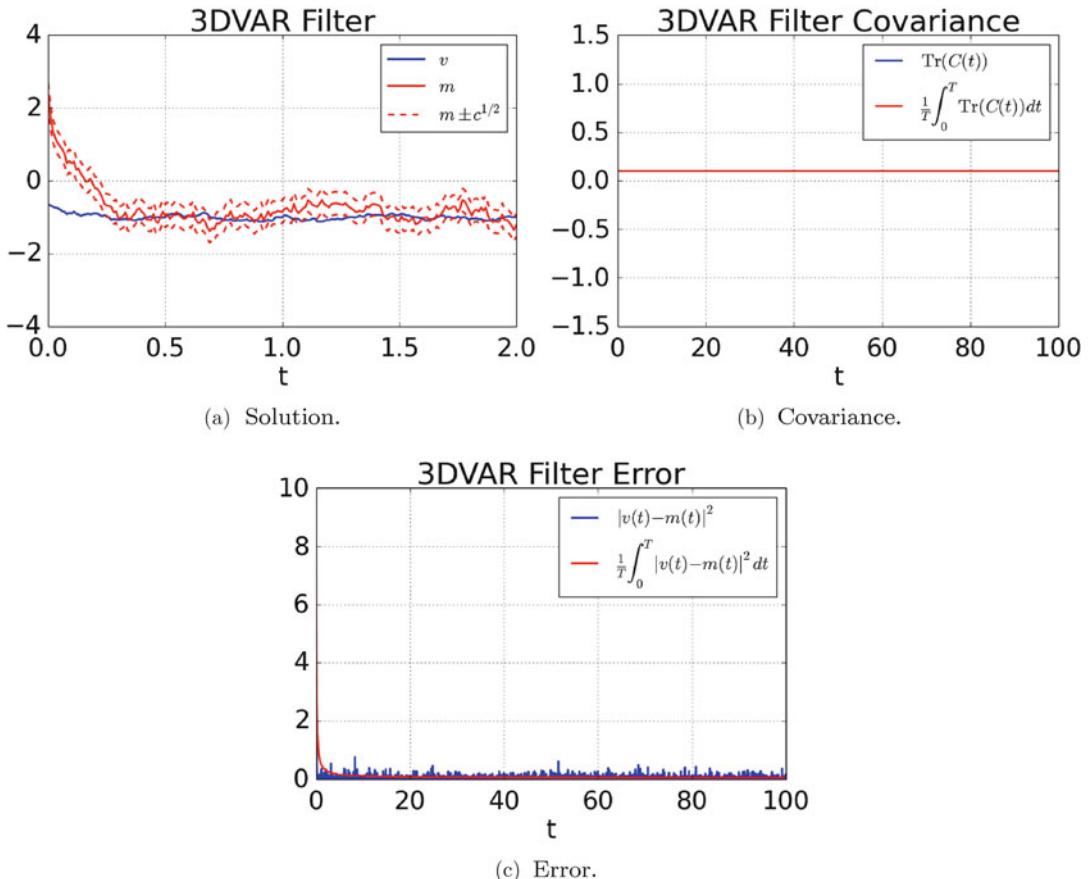


Fig. 8.3: Continuous 3DVAR filter for double well Example 6.5 with $\gamma = 0.1$, $\sigma = 0.3$, as given in Section 9.3.2.

Section 4.4.3. The continuous-time 3DVAR filter acts as a control system, forcing the filter toward the data. In the paper [21], the data for the Navier–Stokes equation comprised low-frequency Fourier information, and this control perspective was generalized in [8] to cover the technically demanding case of data based on pointwise observations. The large-time behavior of the EnKF is studied, in both discrete and continuous time, in [82]. Finally, we note that as discussed in Section 4.6 in the discrete-time setting, the 3DVAR filter may be used to bound the error in the mean of the Kalman filter for linear problems, because of the optimality of the Kalman filter; this latter optimality property follows, as in discrete time, from a Galerkin orthogonality interpretation of the error resulting from taking conditional expectation. The paper [126] implements this idea in the discrete setting.

- Section 8.5 concerns various numerical illustrations. Remark 8.7 highlights the fact that implementing filters in a stable fashion, especially for complex models, can be nontrivial, and the reader is cautioned that blind transfer of the programs in the next chapter to other models may well lead to numerical instabilities. These can be caused by an interaction of the numerical integration method with noisy data. See [59] for a discussion of this.

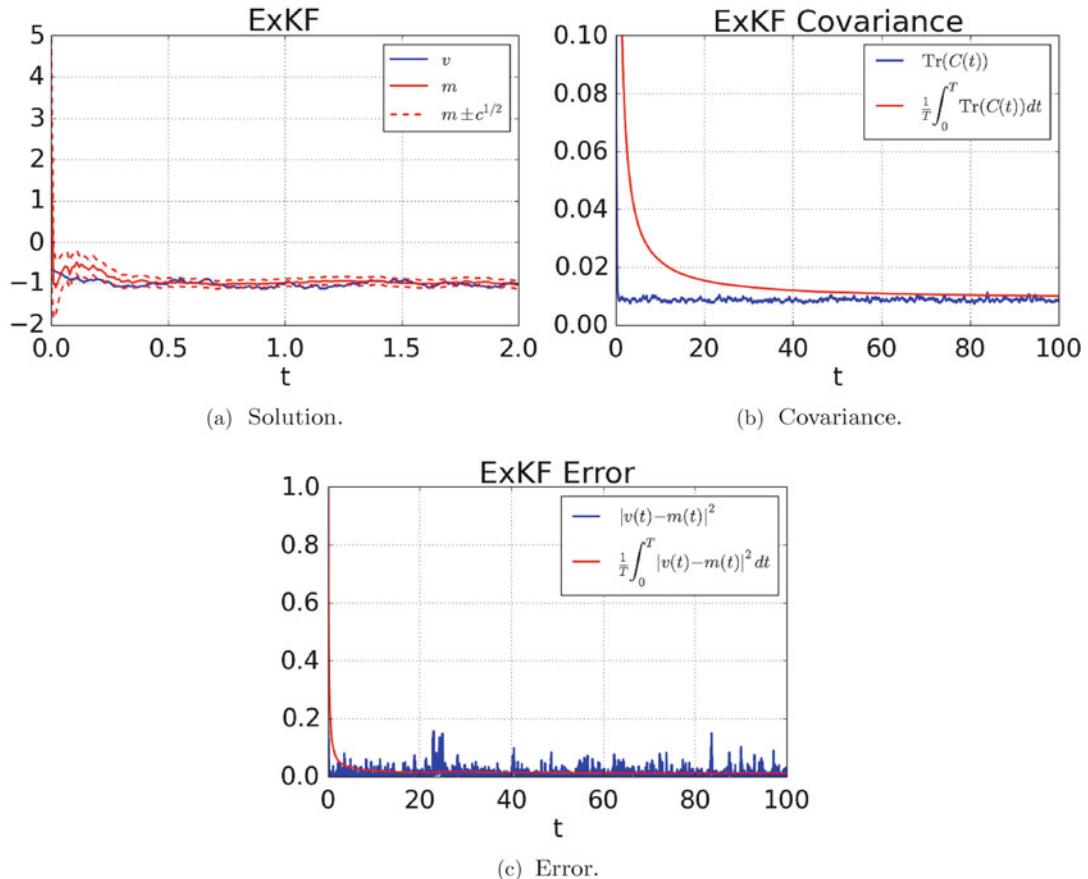


Fig. 8.4: Extended Kalman filter for double-well Example 6.5 with $\gamma = 0.1, \sigma = 0.3$, as given in Section 9.3.3.

8.7 Exercises

1. Consider the linear example of Section 8.4.1. Implement the Kalman–Bucy filter for this problem by modifying program p8c.m. Verify that the code reproduces the large-time asymptotic behavior of the variance as proved in Section 8.4.1. Carefully distinguish between $\ell < 0$, $\ell = 0$, and $\ell > 0$. Now extend your code to include the case $\Sigma_0 = \sigma^2 I$ and study the large-time behavior of the covariance. What can you prove about the large-time behavior of the covariance in this case?
2. In this exercise, we study the properties of the mean for the one-dimensional linear dynamics example considered in Section 8.4.1, in the large-time asymptotic. More specifically, we study the error between the filter and the truth v^\dagger . Assume that the truth satisfies the equation

$$\frac{dv^\dagger}{dt} = \ell v^\dagger,$$

while the data z is given by

$$\frac{dz}{dt} = v^\dagger + \gamma \frac{dB}{dt},$$

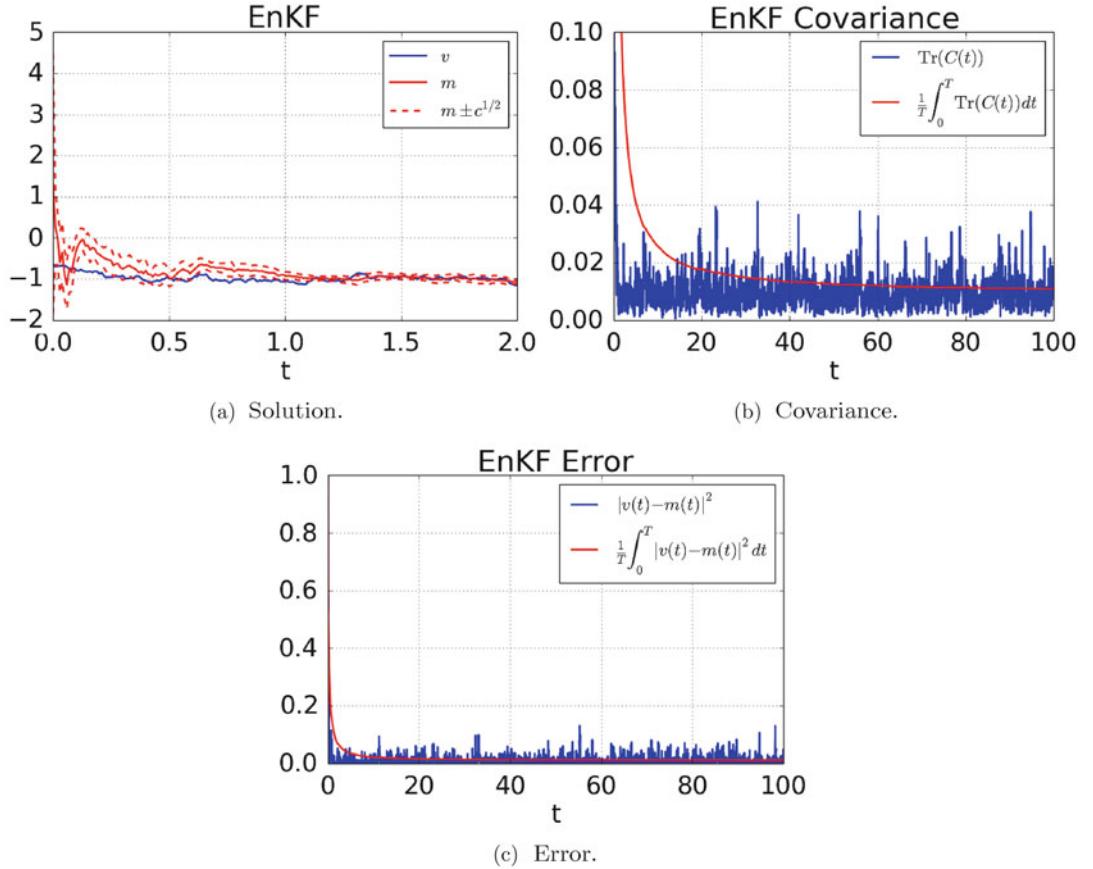


Fig. 8.5: Ensemble Kalman filter for double-well Example 6.5 with $\gamma = 0.1, \sigma = 0.3$, as given in Section 9.3.4.

and B is a realization of the standard unit Brownian motion. Define $e = m - v^\dagger$ and show that

$$\frac{de}{dt} + fe = \gamma^{-1} c \frac{dB}{dt},$$

where $f = F'$ and

$$F'(t) = \gamma^{-2} c(t) - \ell, \quad F(0) = 0.$$

Apply the Itô formula of Lemma 6.3 to a judiciously chosen function to show that

$$e(t) = \exp(-F(t))e(0) + \text{SI}(t), \quad (8.20)$$

where

$$\text{SI}(t) = \int_0^t \gamma^{-1} \exp(F(s) - F(t)) c(s) dB(s).$$

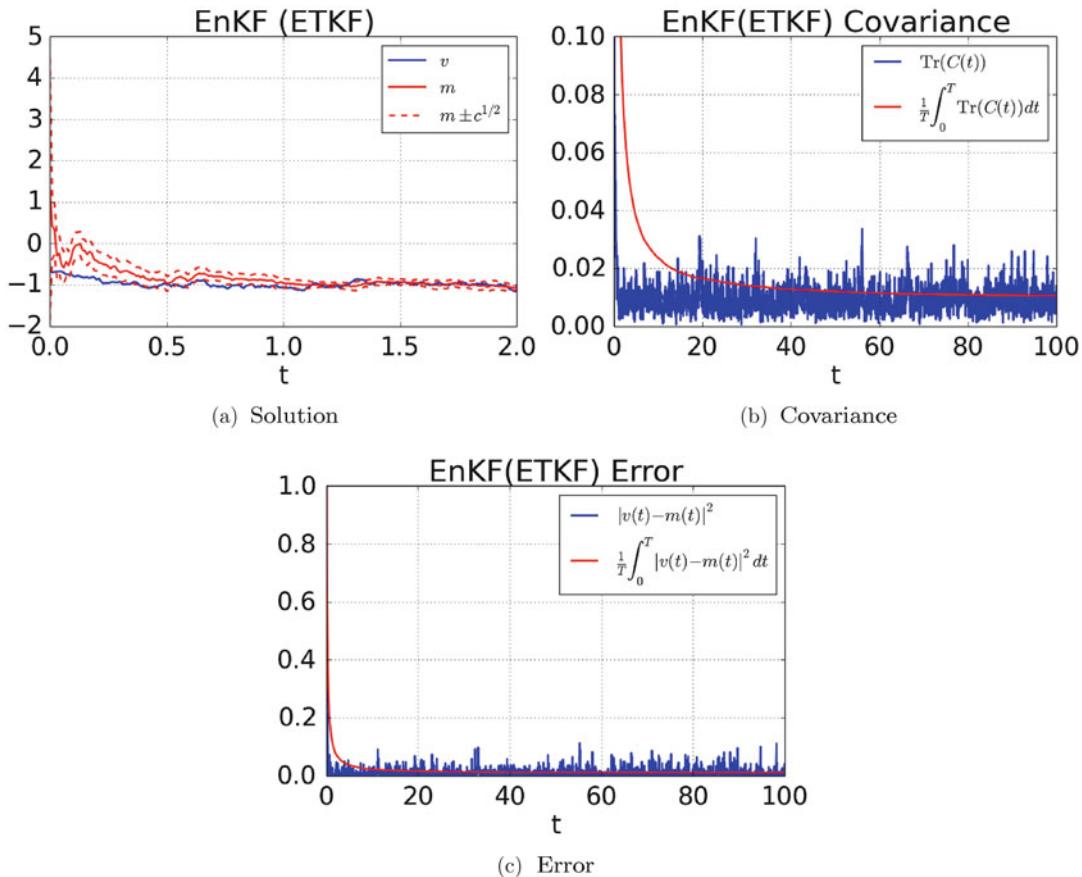


Fig. 8.6: Ensemble transform Kalman filter for double-well Example 6.5 with $\gamma = 0.1$, $\sigma = 0.3$, as given in Section 9.3.5.

Use the Itô isometry of Lemma 6.1(iii) to show that

$$\mathbb{E}|\mathbf{S}(t)|^2 = \int_0^t \gamma^{-2} \exp(2F(s) - 2F(t)) c^2(s) ds. \quad (8.21)$$

Using the properties of the variance established in Section 8.4.1, show that the asymptotic error in the filter mean is bounded by $\mathcal{O}(\gamma^2)$.

- Extend the 3DVAR code of program p10c.m so that it may be applied to the Lorenz '63 example of Example 6.7. Consider both the fully observed case, in which $H = I$, and the partially observed case, with $H = (1, 0, 0,)^T$. Compare the output of the filter with the truth underlying the data, using different observational noise levels.
 - Extend the ExKF code of program p11c.m so that it may be applied to the Lorenz '63 example of Example 6.7. Consider the fully observed case, in which $H = I$. Compare the output of the filter with the truth underlying the data using different observational noise levels.

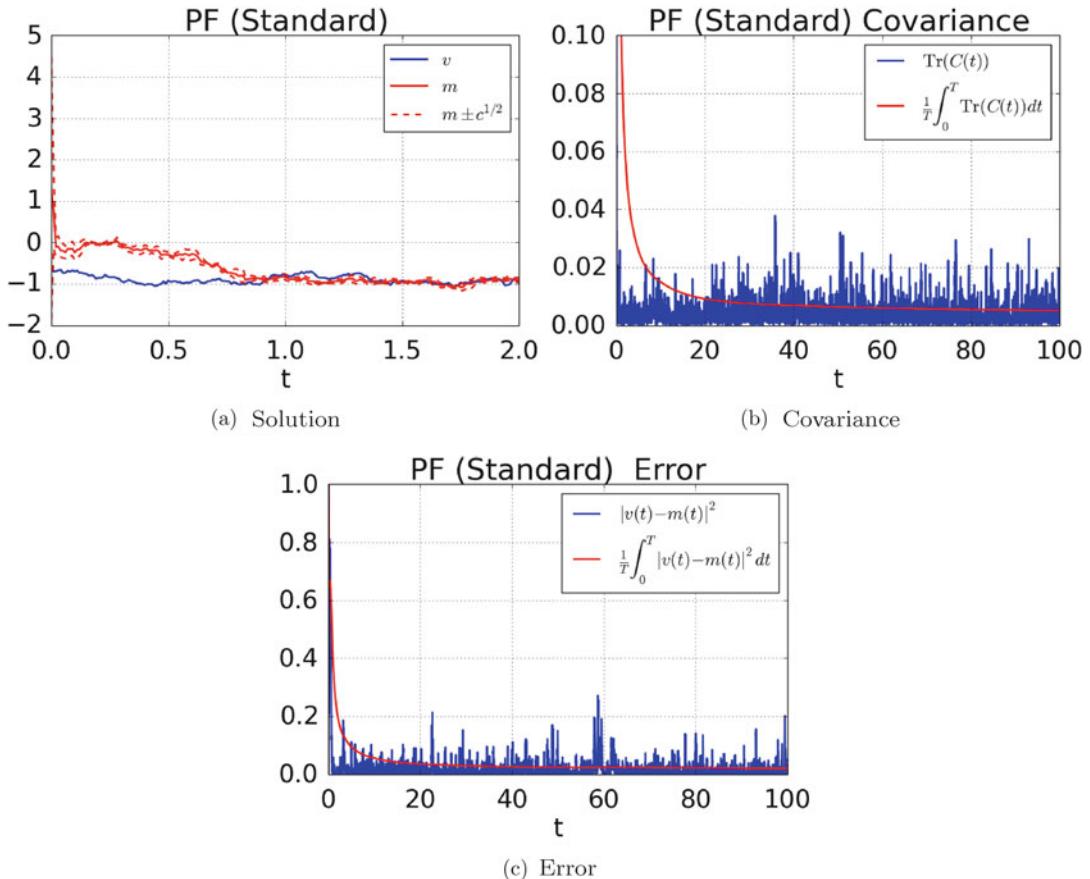


Fig. 8.7: Continuous particle filter (standard) for double-well Example 6.5 with $\gamma = 0.1$, $\sigma = 0.3$, as given in Section 9.3.6.

5. Extend the EnKF code of program p12c.m so that it may be applied to the Lorenz '96 example of Example 6.8. Consider the case in which two out of every three points are observed. Compare the output of the filter with the truth underlying the data, using different observational noise levels.

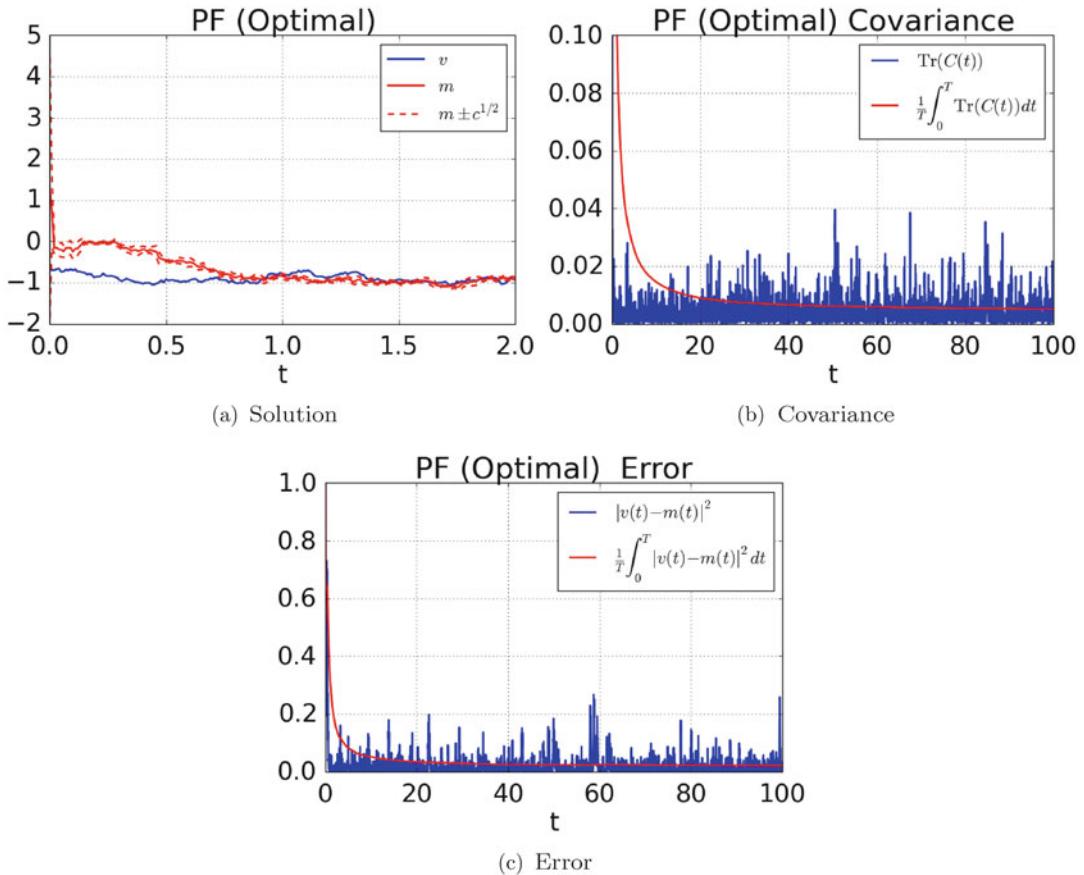


Fig. 8.8: Continuous particle filter (optimal) for double-well Example 6.5 with $\gamma = 0.1$, $\sigma = 0.3$, as given in Section 9.3.7.

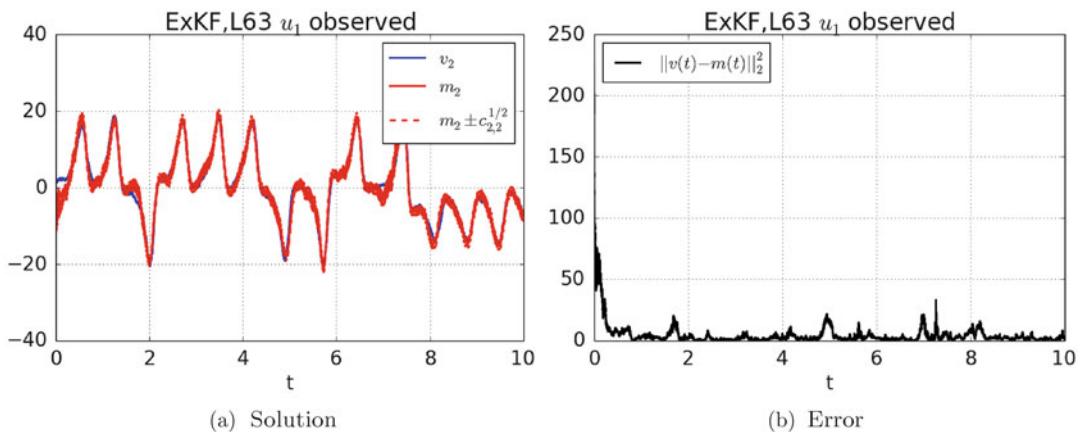


Fig. 8.9: Extended Kalman filter for Lorenz '63 Example 6.7 with observation operator $H = (1, 0, 0)$.

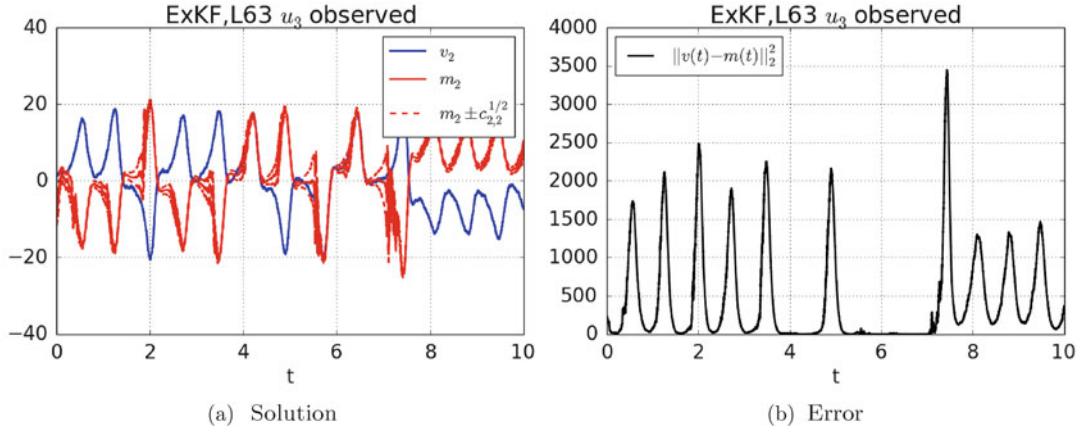


Fig. 8.10: Extended Kalman filter for Lorenz '63 example 6.7 with observation orator $H = (0, 0, 1)$.

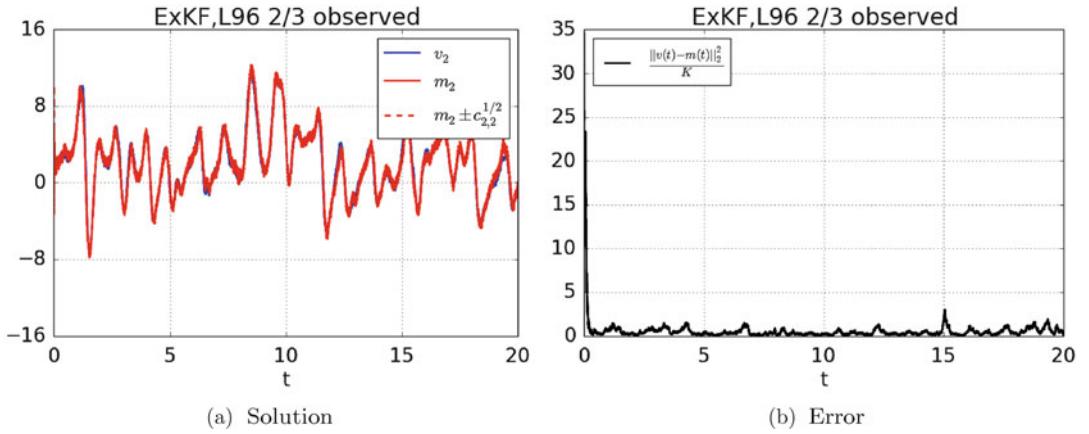


Fig. 8.11: Extended Kalman filter for Lorenz '96 example 6.8 with 2/3 of components observed.

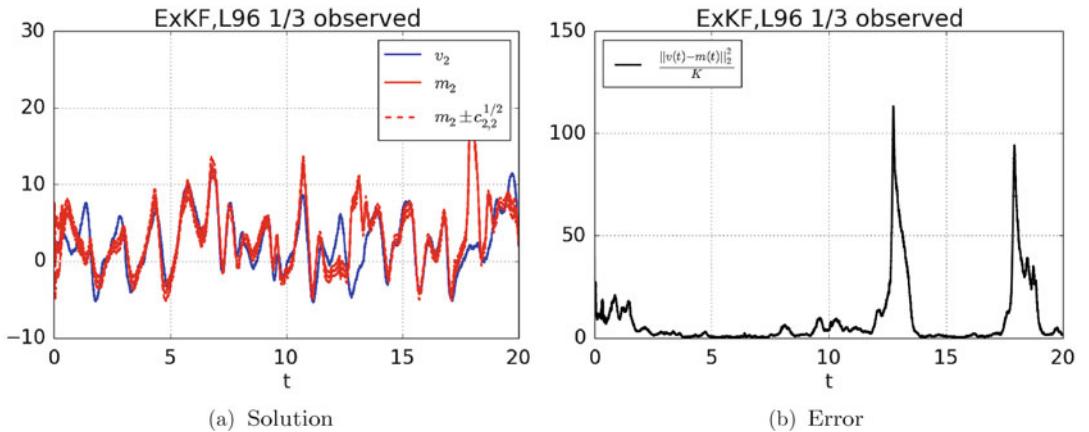


Fig. 8.12: Extended Kalman filter for Lorenz '96 example 6.8 with 1/3 of components observed.

Chapter 9

Continuous Time: MATLAB Programs

This chapter is dedicated to illustrating the examples, theory, and algorithms presented in the preceding three chapters through a few short and easy-to-follow MATLAB programs. We have followed the same principles as in Chapter 9, and again the code may be readily extended to solve problems more complex than those described in Examples 6.4–6.8, which will be used for most of our illustrations.

9.1 Chapter 6 Programs

The programs `p1c.m`, `p2c.m`, and `p3c.m` used to generate the figures in Chapter 6 are presented in this section. These algorithms simply solve the dynamical system (6.4) and process the resulting data.

9.1.1. `p1c.m`

The first program, `p1c.m`, illustrates how to obtain sample paths from equation (6.4). In particular, the program simulates sample paths of the equation

$$\frac{dv}{dt} = v - v^3 + \sqrt{\Sigma_0} \frac{dB_v}{dt}, \quad (9.1)$$

corresponding to Example 6.6, using the Euler–Maruyama discretization. In line 4, the variables `tau` and `T` correspond to the time step and the final time of integration respectively. Furthermore, a vector `t` is also created in this line that stores the time points in which the solution of the SDE is approximated. The parameter `Sigma` is set in line 5, while in line 6, a vector `x` of length `N` (the length of the vector) is created in which the approximation of the SDE will be stored. The seed for the random number generator is set to `sd ∈ N` in line 7, while in line 8, we precalculate the Brownian increments `dW` used in the Euler–Maruyama scheme. This scheme is implemented in lines 11–13, and so the vector `x` now stores the approximation of $v(t)$ at the points given by `t`. Lines 15–22, are used to produce Figure 6.3. Figures 6.1 and 6.2, from Examples 6.4 and 6.5, were obtained by simply modifying lines 11–13 to order to create sample paths corresponding to a different function $f(v)$ in (6.4), where we note that Example 6.4 decouples into two scalar problems.

```

1 clear; set(0,'defaultaxesfontsize',20); format long
2 %% plc.m behaviour of the double well potential with noise
3
4 tau=0.01; T=5e4; t=[0:tau:T];% set up integration constants
5 x0=3; eps=0.08; Sigma=sqrt(2*eps);% set the SDE coefficients
6 N=length(t); x=zeros(1,N); x(1)=x0; % set initial conditions
7 sd=0;rng(sd);% choose random number seed
8 dW=sqrt(tau)*randn(N-1,1);% precalculate the Brownian increments used
9
10 % Euler implementation of the OU
11 for i=1:N-1
12     x(i+1)=x(i)+tau*x(i)*(1-x(i)^2)+Sigma*dW(i);
13 end
14
15 dx=0.01;z=[-5:dx:5]; V=hist(x,z);
16 p=exp(-0.25*eps^-1*(1-z.^2).^2); p1=p/trapz(z,p);
17
18 figure(1), plot(t,x,'k','LineWidth',2)
19 axis([0 1000 -3 3])
20 figure(2), plot(z,V./(dx*sum(V)),'r',z,p1,'k','LineWidth',2)
21 axis([-2 2 0 1.5])
22 legend 'empirical measure' 'invariant measure'

```

9.1.2. p2c.m

The second program, `p2c.m`, is designed to visualize the posterior distribution in the case of linear one-dimensional continuous deterministic dynamics. For clarity, the program is separated into three main sections. The `setup` section in lines 3–8 defines the parameters of the problem. The model parameter λ is defined in line 7; it determines the dynamics of the forward model, in this case given by

$$\frac{du}{dt} = \lambda u. \quad (9.2)$$

We observe that the linearity allows the exact solution $u(t) = u_0 e^{\lambda t}$ to be used in the code. The parameters `m0` and `C0` define the mean and the covariance of the prior distribution $u_0 \sim N(m_0, C_0)$, while `gamma` is the diffusion coefficient for the driving Brownian motion.

The `truth` section in lines 10–19 generates the truth reference trajectory (or truth) `vt` in line 16 using (9.2), as well as the observation `z` found using an Euler–Maruyama discretization of the SDE

$$\frac{dz}{dt} = u(t) + \gamma \frac{dW_z}{dt}. \quad (9.3)$$

The `solution` section after line 21 computes the solution, in this case the pointwise representation of the posterior smoothing distribution on the scalar initial condition. The pointwise values of the initial condition are given by the vector `v0` (u_0) defined in line 21. The corresponding vector of values of `Xi3` (Ξ_3), `Jdet` (J_{det}), and `Idet` (I_{det}) is computed in lines 24, 25, and 27 for each value of `v0`, as related by the equation

$$I_{\text{det}}(v_0; z) = J_{\text{det}}(v_0) + \Xi_3(v_0, z). \quad (9.4)$$

In particular, here we have made explicit use of the fact that $\Psi(v_0; t) = v_0 e^{\lambda t}$ in calculating the first term in the expression for Ξ_3 . The functional $I_{\text{det}}(v_0; z)$ is the negative log-posterior

as given in Theorem 6.15. Having obtained $I_{\text{det}}(v_0; z)$, we calculate $\mathbb{P}(v_0|z)$ in lines 30–32 using the formula

$$\mathbb{P}(v_0|z) = \frac{\exp(-I_{\text{det}}(v_0; z))}{\int \exp(I_{\text{det}}(v_0; z))}.$$

The rest of the program deals with plotting our results, and in this instance, it coincides with the curve shown for $T = 10^2$ in Figure 6.7a. Simple modifications of this program were used to produce the rest of Figure 6.7.

```

1 clear;set(0,'defaultaxesfontsize',20);format long
2 % p2c.m - smoothing problem for continuous time OU process
3 %% setup
4 C0=5;m0=4;% variance and mean of the prior
5 sd=1;rng(sd);% choose random number seed
6 T=10; tau=0.01; t=[0:tau:T]; N=length(t);% time discretization
7 lambda=-0.5;% dynamics determined by lambda
8 gamma=1;% observational noise variance is gamma^2
9
10 %% truth
11 vt=zeros(N,1); z=zeros(N,1);% preallocate space to save time
12 vt(1)=0.5; z(1)=0;% truth initial condition
13 dW=sqrt(tau)*randn(N-1,1);% precalculating the Brownian increments used
14 for i=1:N-1
15     % can be replaced Psi for each problem
16     vt(i+1)=exp(lambda*tau)*vt(i);% create truth
17     z(i+1)=z(i)+tau*vt(i)+gamma*dW(i);% create data
18 end
19
20 %% solution
21 v0=[-10:0.01:10];% construct vector of different initial data
22 Xi3=zeros(length(v0),1); Idet=Xi3; Jdet=Xi3;% preallocate space to save
23 time
24 for j=1:length(v0)
25     Jdet(j)=1/2/C0*(v0(j)-m0)^2;% background penalization
26     Xi3(j)=1/2*v0(j)^2*gamma^-2*(exp(2*lambda*T)-1)/(2*lambda) - ...
27         sum(v0(j)*exp(lambda*t(1:end-1)).*diff(z'))/gamma^2;
28     Idet(j)=Xi3(j)+Jdet(j);
29 end
30
31 constant=trapz(v0,exp(-Idet));% approximate normalizing constant
32 P=exp(-Idet)/constant;% normalize posterior distribution
33 prior=normpdf(v0,m0,C0); % calculate prior distribution
34
35 figure(1),plot(v0,prior,'k','LineWidth',2)
36 hold on, plot(v0,P,'r--','LineWidth',2), xlabel 'v_0',
37 legend 'prior' T=10^2

```

9.1.3. p3c.m

The third program, p3c.m, is used to visualize the posterior in the case of the double-well potential model (9.1) in the case of no noise $\Sigma_0 = 0$. The main difference from p2.m is that in this case, an explicit solution is not used (although it can be computed exactly; see the exercises from Chapter 7), and we approximate $\Psi(v_0; t)$ using an Euler approximation in line

18. This fact is also taken into account when Ξ_3 (Xi3) is calculated in line 31. The program ends by plotting our results, and in this instance, it coincides with the line for $T = 10$ in Figure 6.9a. Simple modifications of this program were applied to the remainder of Figure 6.9, as well as Figure 6.8.

```

1 clear;set(0,'defaulaxesfontsize',20);format long
2 %% p3c.m - smoothing problem for continuous time double-well
3 %% setup
4
5 C0=5;% variance of the prior
6 m0=-4;% mean of the prior
7 sd=1;rng(sd);% choose random number seed
8 T=10; tau=0.01; t=[0:tau:T]; N=length(t);% time discretization
9 gamma=1;% observational noise variance is gamma^2
10
11 %% truth
12
13 vt=zeros(N,1); z=zeros(N,1);% preallocate space to save time
14 vt(1)=0.5; z(1)=0;% truth initial condition
15 dW=sqrt(tau)*randn(N-1,1);% precalculate the Brownian increments used
16 for i=1:N-1
17     % can be replaced Psi for each problem
18     vt(i+1)=vt(i)+tau*(vt(i)-vt(i)^3);
19     z(i+1)=z(i)+tau*vt(i)+gamma*dW(i);% create data
20 end
21
22 %% solution
23
24 v0=[-10:0.01:10];% construct vector of different initial data
25 Xi3=zeros(length(v0),1); Idet=Xi3; Jdet=Xi3;% preallocate space to save
26 time
27 for j=1:length(v0)
28     vv=zeros(N,1); vv(1)=v0(j);
29     Jdet(j)=1/2/C0*(v0(j)-m0)^2;% background penalization
30     for i=1:N-1
31         vv(i+1)=vv(i)+tau*(vv(i)-vv(i)^3);
32         Xi3(j)=Xi3(j)+(tau*0.5*vv(i)^2-vv(i)*(z(i+1)-z(i)))/gamma^2;
33     end
34     Idet(j)=Jdet(j)+Xi3(j);
35 end
36
37
38 constant=trapz(v0,exp(-Idet));% approximate normalizing constant
39 P=exp(-Idet)/constant;% normalize posterior distribution
40 prior=normpdf(v0,m0,C0); % calculate prior distribution
41
42 figure(1),plot(v0,prior,'k','LineWidth',2)
43 hold on, plot(v0,P,'r--','LineWidth',2), xlabel 'v_0',
44 legend 'prior' T=10

```

9.2 Chapter 7 Programs

The programs `p4c.m`-`p7c.m` used to generate the figures in Chapter 7 are presented in this section. Hence various MCMC algorithms used to sample the posterior distribution are given,

together with a variational algorithm. Similarly to the discrete-time case, we have tried to be consistent in our notation.

9.2.1. p4c.m

The MATLAB program `p4c.m` is the first of the MCMC algorithms discussed in continuous time. It contains an implementation of the random walk Metropolis (RWM) algorithm from Section 7.2.1 to determine the posterior distribution on the initial condition arising from the double-well Example 6.6 with $\epsilon = 0$, and equation (9.1) in particular. Note that in this case, since the underlying dynamics are deterministic and hence completely determined by the initial condition, the RWM algorithm will provide samples from a probability distribution on \mathbb{R} . As in program `p2c.m`, the code is divided into three sections: `setup`, where parameters are defined, `truth`, where the truth and data are generated, and `solution`, where the solution is computed, this time by means of MCMC samples from the posterior smoothing distribution. The parameters are set in lines 5–9, and the true solution (here taken as only the initial condition, rather than the trajectory it gives rise to) `vt` is calculated in line 18. The true value `vt` is also used as the initial sample in the Markov chain for this and for all subsequent MCMC programs. This scenario is not possible in the case that the data is not simulated. However, it is useful in the case that the data is simulated as it is here, because it reduces the time necessary for the current sample in the chain to reach the target distribution or the high probability region of the state space. Therefore, the value of $I_{\text{det}}(v^\dagger)$, denoted by the temporary variable `Idet`, will be necessary to compute the acceptance probability, as described below. It is computed in lines 15–22 exactly as in lines 25–32 of program `p3c.m`, as described around (9.4). In the `solution` section, some additional MCMC parameters are defined. In line 27, the number of samples is set to `M=10^5`. For the parameters and specific data used here, this is sufficient for the convergence of the Markov chain. In line 29, the step-size parameter `beta` is preset such that the algorithm for this particular posterior distribution has a reasonable acceptance probability, or ratio of accepted vs. rejected moves. The vector `V` defined in line 28 will save all the samples. In line 33, a move is proposed according to the proposal equation

$$w^{(k)} = v^{(k-1)} + \beta \iota^{(k-1)},$$

where $v(v)$ is the current state of the chain (initially taken to be equal to the true initial condition v_0), $\iota^{(k-1)}=\text{randn}$ is an i.i.d. standard normal, and w represents $w^{(k)}$. Indices are not used for `v` and `w`, because they will be overwritten at each iteration.

The temporary variable `vv` is again used for the trajectory corresponding to $w^{(k)}$ as a vehicle to compute the value of the proposed $I_{\text{det}}(w^{(k)}; y)$, denoted in line 41 by `Idetprop=Jdetprop+Xi3prop`. This is later used in lines 43–45, where we decide to accept or reject the proposal according to the acceptance probability

$$a(v^{(k-1)}, w^{(k)}) = 1 \wedge \exp(I_{\text{det}}(v^{(k-1)}; y) - I_{\text{det}}(w^{(k)}; y)).$$

The rest of the program (lines 50–51) uses the samples stored in `V` to visualize the posterior distribution. The output is then compared with the corresponding output of `p3c.m` for the same parameters in Figure 6.8a.

```

1 clear; set(0,'defaultaxesfontsize',20); format long
2 %% p4c.m MCMC RWM algorithm for double well (Ex. 5.6)
3 %% setup
4
5 C0=5;% variance of the prior
6 m0=4;% mean of the prior
7 sd=1;rng(sd);% choose random number seed
8 T=1; tau=0.1; t=[0:tau:T]; N=length(t);% time discretization
9 gamma=1;% observational noise variance is gamma^2
10
11 %% truth
12 vt(1)=0.5; z(1)=0;% truth initial condition
13 dW=sqrt(tau)*randn(N-1,1);% precalculate the Brownian increments used
14 Jdet=1/2/C0*(vt(1)-m0)^2;% background penalization
15 Xi3=0;% initialization model-data misfit functional
16 for i=1:N-1
17     % can be replaced Psi for each problem
18     vt(i+1)=vt(i)+tau*(vt(i)-vt(i)^3);
19     z(i+1)=z(i)+tau*vt(i)+gamma*dW(i); % create data
20     Xi3=Xi3+(tau*0.5*vt(i)^2-vt(i)*(z(i+1)-z(i)))/gamma^2;
21 end
22 Idet=Jdet+Xi3;% compute log posterior of the truth
23
24 %% solution
25 % Markov Chain Monte Carlo: N forward steps of the
26 % Markov Chain on R (with truth initial condition)
27 M=1e5;% number of samples
28 V=zeros(M,1);% preallocate space to save time
29 beta=0.5;% step-size of random walker
30 v=vt(1);% truth initial condition (or else update I0)
31 n=1; bb=0; rat(1)=0;
32 while n<=M
33     w=v+sqrt(2*beta)*randn;% propose sample from random walker
34     vv(1)=w;
35     Jdetprop=1/2/C0*(w-m0)^2;% background penalization
36     Xi3prop=0;
37     for i=1:N-1
38         vv(i+1)=vv(i)+tau*(vv(i)-vv(i)^3);
39         Xi3prop=Xi3prop+(tau*0.5*vv(i)^2-vv(i)*(z(i+1)-z(i)))/gamma^2;
40     end
41     Idetprop=Jdetprop+Xi3prop;% compute log posterior of the proposal
42
43     if rand<exp(Idet-Idetprop)% accept or reject proposed sample
44         v=w; Idet=Idetprop; bb=bb+1;% update the Markov chain
45     end
46     rat(n)=bb/n;% running rate of acceptance
47     V(n)=v;% store the chain
48     n=n+1 ;
49 end
50 dx=0.05; v0=[-10:dx:10]; Z=hist(V,v0);% construct the posterior histogram
51 figure(1), plot(v0,Z/trapz(v0,Z),'k','Linewidth',2)% visualize the posterior

```

9.2.2. p5c.m

The MATLAB program `p5c.m` contains an implementation of the independence dynamics sampler for stochastic dynamics, as introduced in Section 7.2.2. Thus the posterior distribution is on the entire signal on $L^2([0, T]; \mathbb{R})$. The forward model in this case uses equation (9.1), but now with $\epsilon > 0$. In practice, we cannot sample from $L^2([0, T]; \mathbb{R})$, since it is infinite-dimensional, so we have to discretize. In particular, we choose our time step τ in line 9 and take $T = 10$, and thus the smoothing distribution $\mathbb{P}(v|z)$ is over the state space \mathbb{R}^J (with J defined in line 12).

The sections `setup`, `truth`, and `solution` are defined as for program `p4c.m`. Since the state space is now the pathspace, rather than the initial condition (analogously to the discrete-time program of Section 5.2.1), the truth $vt \in \mathbb{R}^J$ is now a vector. Its initial condition is taken as a draw from $N(m_0, C_0)$ in line 16, and the trajectory is computed in line 20, so that at the end, $vt \sim \rho_0$. Again, v^\dagger (vt) will be the initial condition in the Markov chain, and so $\Xi_2(v^\dagger; z)$ is computed in line 27. Recall from Section 7.2.2 that only $\Xi_2(\cdot; y)$ is required to compute the acceptance probability in this algorithm.

The current state of the chain $v^{(k)}$ and the value of $\Xi_2(v^{(k)}; y)$ are again denoted by `v` and `Xi2`, while the proposal $w^{(k)}$ and the value of $\Xi_2(w^{(k)}; y)$ are again denoted by `w` and `Xi2prop`, as in program `p4c.m`. As discussed in Section 7.2.2, the proposal $w^{(k)}$ is an independent sample from the prior distribution ρ_0 , similarly to v^\dagger , and it is constructed in lines 38–42. The acceptance probability used in line 44 is now

$$a(v^{(k-1)}, w^{(k)}) = 1 \wedge \exp(\Xi_2(v^{(k-1)}; y) - \Xi_2(w^{(k)}; y)). \quad (9.5)$$

The rest of the program is structured similarly to that in Section 9.2.1. The outputs of this program are used to plot Figures 7.3, 7.4, and 7.5. Note that in the case of Figure 7.5, we have used $N = 10^6$ samples.

```

1 clear; set(0,'defaultaxesfontsize',20); format long
2 %% p5c.m MCMC INDEPENDENCE DYNAMICS SAMPLER algorithm in continuous time
3 %% for double well (Ex 5.6) with noise
4
5 %% setup
6 m0=0;% prior initial condition mean
7 C0=1;% prior initial condition variance
8 gamma=0.5; % observational noise variance is gamma^2
9 tau=0.1; T=1;% timestep and final time of integraion
10 epsilon=0.08; sigma=sqrt(2*epsilon);% dynamics noise variance is sigma^2
11 sd=1;rng(sd);% Choose random number seed
12 t=[0:tau:T]; J=length(t);% number of points where we approximate u
13
14 %% truth
15 vt=zeros(J,1); z=zeros(J,1); dz=zeros(1,J-1);% preallocate space to save
16 time
17 ut=sqrt(C0)*randn; z(1)=0;
18 dW_v=sqrt(tau)*randn(J,1);% truth noise sequence model
19 dW_z=sqrt(tau)*randn(J,1);% truth noise sequence observation
20 vt(1)=ut(1);% truth initial condition
21 Xi2=0;
22
23 for j=1:J-1
24     vt(j+1)=vt(j)+tau*vt(j)*(1-vt(j)^2)+sigma*dW_v(j);% create truth
25     z(j+1)=z(j)+tau*vt(j)+gamma*dW_z(j);% create data
26     dz(j)=z(j+1)-z(j);
27     % calculate Xsi_2(v;z) from (5.27)
28     Xi2=Xi2+(0.5*tau*vt(j)^2-vt(j)*dz(j))/gamma^2;
29 end
30
31 %% solution
32 % Markov Chain Monte Carlo: N forward steps of the
33 % Markov Chain on R^{J+1} with truth initial condition
34 N=1e4;% number of samples
35 V=zeros(N,J);% preallocate space to save time
36 v=vt;% truth initial condition
37 n=1; bb=0; rat(1)=0;
38 while n<=N
39     w(1)=sqrt(C0)*randn;% propose sample from the prior distribution
40     Xi2prop=0;
41     for j=1:J-1
42         w(j+1)=w(j)+tau*w(j)*(1-w(j)^2)+sigma*sqrt(tau)*randn;
43         Xi2prop=Xi2prop+(0.5*tau*w(j)^2-w(j)*dz(j))/gamma^2;
44     end
45     if rand<exp(Xi2-Xi2prop)% accept or reject proposed sample
46         v=w; Xi2=Xi2prop; bb=bb+1;% update the Markov chain
47     end
48     rat(n)=bb/n;% running rate of acceptance
49     V(n,:)=v;% store the chain
50     n=n+1;
51 end
52 % plot acceptance ratio and cumulative sample mean
53 figure;plot(rat);figure;plot(cumsum(V(1:N,end))./[1:N]')
54 xlabel('samples N');ylabel('(1/N) \Sigma_{n=1}^N v_0^{(n)}')
55 figure; plot([1:1:N],V(:,end))

```

9.2.3. p6c.m

The independence dynamics sampler of Section 9.2.2 can be very inefficient, since typical random draws from the dynamics can form a poor fit to the data, in comparison with a current state that has a good fit, and will then be rejected. The sixth MATLAB program, p6c.m, gives an implementation of the pCN algorithm from Section 7.2.2, which is designed to overcome this issue by including the parameter β , which, if chosen small, allows for incremental steps in signal space and hence the possibility of nonnegligible acceptance probabilities. This program is used to generate Figure 7.6

This program is almost identical to p5c.m, and so only the points at which it differs will be described. First, since the acceptance probability is given by

$$a(v^{(k-1)}, w^{(k)}) = 1 \wedge \exp(\Xi_2(v^{(k-1)}; y) - \Xi_2(w^{(k)}; y) + \Xi_1(v^{(k-1)}) - \Xi_1(w^{(k)})),$$

the quantity

$$\Xi_1(v) := \frac{1}{2} \int_0^T |f(v(t))|_{\Sigma_0}^2 dt - \int_0^T \langle f(v(t)), dv(t) \rangle_{\Sigma_0}$$

will need to be computed, both for $v^{(k)}$, denoted by v in lines 32 and 47, where its value is denoted by $\text{Xi1}(v^{(0)} = v^\dagger)$ and $\Xi_1(v^\dagger)$ is computed in line 25), and for $w^{(k)}$, denoted by w in line 39, where its value is denoted by Xi1prop in line 43.

As discussed in Section 7.2.2, the proposal $w^{(k)}$ is given by

$$w^{(k)} = m + (1 - \beta^2)^{\frac{1}{2}}(v^{(k-1)} - m) + \beta \xi^{(k-1)}; \quad (9.6)$$

here $\xi^{(k-1)}$ is a standard Brownian motion on \mathbb{R} starting from an initial condition drawn at random from $N(0, C_0)$. We denote this by xi in line 38.

```

1 clear; set(0,'defaultaxesfontsize',20); format long
2 %% p6c.m MCMC pCN algorithm algorithm in continuous time
3 %% for double well (Ex 5.6) with noise
4 %% setup
5 m0=0;% prior initial condition mean
6 C0=1;% prior initial condition variance
7 gamma=0.5;% observational noise variance is gamma^2
8 tau=0.1; T=1;% timestep and final time of integration
9 epsilon=0.08; sigma=sqrt(2*epsilon);% dynamics noise variance is sigma^2
10 sd=1;rng(sd);% Choose random number seed
11 t=[0:tau:T]; J=length(t);% number of points where we approximate u
12 %% truth
13 vt=zeros(1,J); z=zeros(1,J); dz=zeros(1,J-1); dvt=zeros(1,J-1);%
14 preallocate
15 ut=[sqrt(C0)*randn,sigma*sqrt(tau)*randn(1,J-1)];% truth noise sequence
16 z(1)=0;dW_z=sqrt(tau)*randn(1,J-1);% truth noise sequence observation
17 vt(1)=ut(1);% truth initial condition
18 X12=0; X11=0;
19 for j=1:J-1
20     vt(j+1)=vt(j)+tau*vt(j)*(1-vt(j)^2)+ut(j+1); % create truth
21     dvt(j)=vt(j+1)-vt(j);
22     z(j+1)=z(j)+tau*vt(j)+gamma*dW_z(j); % create data
23     dz(j)=z(j+1)-z(j);
24     % calculate Xsi_2(v,z) from (5.27)
25     X12=X12+(0.5*tau*vt(j)^2-vt(j)*dz(j))/gamma^2;
26     X11=X11+(0.5*tau*(vt(j)*(1-vt(j)^2))^2- vt(j)*(1-vt(j)^2)*dvt(j))/
27         /sigma^2;
28 end
29 %% solution
30 % Markov Chain Monte Carlo: N forward steps of the
31 % Markov Chain on R^{J+1} with truth initial condition
32 N=1e5;% number of samples
33 V=zeros(N,J);% preallocate space to save time
34 v=vt;% truth initial condition
35 beta=0.05;% step-size of pCN walker
36 n=1; bb=0; rat(1)=0;
37 m=[m0,zeros(1,J-1)];
38 while n<=N
39     dW=[sqrt(C0)*randn,sigma*sqrt(tau)*randn(1,J-1)];% Brownian increments
40     xi=cumsum(dW); % Brownian motion starting at a random initial condition
41     w=m+sqrt(1-beta^2)*(v-m)+beta*xi;% propose sample from the pCN walker
42     Xi2prop=0; X11prop=0;
43     for j=1:J-1
44         Xi2prop=Xi2prop+(0.5*tau*w(j)^2-w(j)*dz(j))/gamma^2;
45         X11prop=X11prop+0.5*tau*(w(j)*(1-w(j)^2))^2/sigma^2- ...
46         w(j)*(1-w(j)^2)*(w(j+1)-w(j))/sigma^2;
47     end
48     if rand<exp(Xi2-Xi2prop+X11-X11prop)% accept or reject proposed sample
49         v=w; Xi2=Xi2prop; X11=X11prop; bb=bb+1;% update the Markov chain
50     end
51     rat(n)=bb/n;% running rate of acceptance
52     V(n,:)=v;% store the chain
53     n=n+1;
54 end
55 % plot acceptance ratio and cumulative sample mean
56 figure;plot(rat);figure;plot(cumsum(V(1:N,:))./[1:N]');
```

9.2.4. p7c.m

The program `p7c.m` presents the MAP optimization algorithm as applied to the deterministic Lorenz '63 model 6.23 with continuous observations. As usual, the parameters are defined in the `setup` section of lines 4–14. Similarly to `p16.m`, which implements the deterministic Lorenz '63 model (here the case $\sigma = 0$), the program is written as a function rather than a script, and again, an auxiliary function, `f`, is defined to evaluate the right-hand side of the ODE, on line 48. The Euler approximation of the true signal v^\dagger and the observation dz^\dagger are defined in lines 19 and 20, respectively. The MATLAB built-in function `fminsearch` is used here again on line 33, similarly to `p7.m`, for the optimization of the auxiliary function I as a function of its argument u . The auxiliary function here is I_{det} from (6.35)a), defined on lines 40–46. This is a sum of J_{det} from (6.35)b), defined on line 41, and Ξ_3 from (6.35)(c), defined on line 44 within the loop of lines 42–45, where the path associated to the unknown input u is defined on line 43 also within the loop. The output `out` of the function can be recognized as I_{det} from (6.35). On line 25, one may choose the random number seed `sd` to begin with a different random initial condition `uu` drawn from the prior on line 28. One may also uncomment line 29 to try the optimization beginning with the truth as initial condition. Lines 35–37 plot the results for a single initial condition. The results of the top left panel of Fig. 7.1 can be reproduced by setting `J=2e4` on line 6 of this program. The map estimator is found by setting the initial condition to `uu=vt(:,1)` on line 29, or by letting the seed `sd` on line 25 be given as 1 or 100, for example. The other two minima may be found by setting the seed value to 10 or 1000, respectively. A simple modification can be used to loop through the conditionals and generate the other three panels of that figure.

```

1 function this=p7c
2 clear;set(0,'defaultaxesfontsize',20);format long
3 %% p7c.m 4DVAR for inverse Lorenz '63
4 %% setup
5
6 J=0.5e4;% number of steps
7 a=10;b=8/3;r=28;% define parameters
8 gamma=1e-1;% observational noise variance is gamma^2
9 C0=1e-2*eye(3);% prior initial condition covariance
10 m0=zeros(3,1);% prior initial condition mean
11 sd=1;rng(sd);% choose random number seed
12 H=[1,0,0];% observation operator
13 tau=1e-4;
14
15 %% truth
16
17 vt(:,1)=m0+sqrtm(C0)*randn(3,1);% truth initial condition
18 for j=1:J
19     vt(:,j+1)=vt(:,j) + tau*f(vt(:,j),a,b,r);% create truth
20     dz(:,j)=tau*H*vt(:,j+1)+gamma*sqrt(tau)*randn;% create data
21 end
22
23 %% solution
24
25 sd=1;rng(sd);% try changing the seed for different
26             % initial conditions -- if the result is not the same,
27             % there may be multimodality.
28 uu=sqrtm(C0)*randn(3,1);% initial guess
29 %uu=vt(:,1);           % truth initial guess option
30
31 % solve with blackbox
32 % exitflag=1 ==> convergence
33 [vmap,fval,exitflag]=fminsearch(@(u) I(u,dz,gamma,m0,C0,J,H,a,b,r,tau),uu)
34
35 figure;plot(vmap,'ko','Markersize',20,'Linewidth',2);%axis([0 4 -1 1.5]);
36 hold;plot(vt(:,1),'rx','Markersize',20,'Linewidth',2);
37 hold;xlabel('u');legend('MAP','truth')
38
39 %% auxiliary objective function definitions
40 function out=I(u,dz,gamma,m0,C0,J,H,a,b,r,tau)
41 Jdet=1/2*(u-m0)'*(C0\ (u-m0));Xi3=0;v=zeros(3,J);v(:,1)=u;
42 for j=1:J
43     v(:,j+1)=v(:,j)+tau*f(v(:,j),a,b,r);
44     Xi3=Xi3+1/gamma^2*(norm(H*v(:,j+1))^2/2*tau - dz(j)'*H*v(:,j+1));
45 end
46 out=Xi3+Jdet;
47
48 function rhs=f(y,a,b,r)
49 rhs(1,1)=a*(y(2)-y(1));
50 rhs(2,1)=-a*y(1)-y(2)-y(1)*y(3);
51 rhs(3,1)=y(1)*y(2)-b*y(3)-b*(r+a);

```

9.3 Chapter 8 Programs

This section consists of the programs `p8c.m` through `p17c.m`, which were used to generate the illustrations in Chapter 8, where filtering algorithms are considered. It is worth noting that the filtering algorithms themselves are continuous-time dynamical systems with very particular structures, which needs to be preserved by the numerical method one uses to implement them. For example, the covariance must remain symmetric and positive definite. While standard discretization methods will often destroy these important structural properties, implementing the specific discretization arising from the discrete-time formulation of the filtering algorithm preserves the desired properties. Therefore, the filtering algorithms presented in this section are very similar to the filtering algorithms in Section 5.3, with appropriate scalings with respect to time step τ as given in equation (6.1). There are some areas where important differences with discrete time remain; these will be described in what follows.

9.3.1. p8c.m

Program `p8c.m` solves the continuous-time Kalman–Bucy filter, analogous to the discrete-time case in `p8.m`. The discretization time step `tau` is set in line 14. The linear forward SDE is

$$dv = Avdt + \sqrt{\Sigma_0}dB_v.$$

The matrix

$$A = \begin{pmatrix} -1 & 1 \\ -1 & -1 \end{pmatrix}$$

is defined on line 13 and has eigenvalues with negative real part, so that the dynamics are contractive. The values $\Sigma_0 = \sigma^2 I$ and σ are defined on line 9. Notice that for a continuous-time linear equation with additive noise, the discretized form with time step τ is identical to that in the discrete-time case, except with the forward operator now given as the exponential of an operator $L = e^{A\tau}$, defined in line 15. The dynamical noise covariance of the corresponding discrete system is then given in the form $\Sigma = \sigma^2(A + A^\top)^{-1}(e^{(A+A^\top)\tau} - I)$. This is defined in line 16, and its square root in line 17 for use below. In line 19, the vector `z` is defined and preallocated, with the default initial condition `z(1)=0` prescribed in line 23.

The dynamical evolution of the hidden process `v` and that of the observed process `z` are given in lines 29–30. The former looks just like its analogue in the discrete case, while the latter is slightly different, since the noisy observation is not of `y(j)` here, but of the increment `dz`, given by `z(j+1)-z(j)`.

A split-step scheme is used for incorporation of the observations using the update step of the discrete equations from which the continuous-time equations were derived, as explained in the introduction. Therefore, there are intermediate “predict” mean and covariance `mhat` and `chat`, as in the discrete case, in lines 32–33. The corresponding innovation and Kalman gain are then defined in lines 35–36, and finally, the update completes the time step forward of size `tau` of the Kalman–Bucy filter in lines 37–38. That is, lines 37–38 complete the approximation of the equations for the mean and covariance given in Theorem 8.1, with L replaced by A . As in the discrete-time examples, the resulting trajectory, covariance evolution, and error are plotted in the remaining lines. The results are presented in Fig. 8.1.

```

1 clear;set(0,'defaultaxesfontsize',20);format long
2 %% p8c.m Kalman-Bucy Filter
3 %% setup
4
5 J=1e4;% number of steps
6 N=2;%dimension of state
7 I=eye(N);% identity operator
8 gamma=1;% observational noise variance is gamma^2*I
9 sigma=1;% dynamics noise variance is sigma^2*I
10 C0=eye(2);% prior initial condition variance
11 m0=[0;0];% prior initial condition mean
12 sd=10;rng(sd);% choose random number seed
13 A=[-1,-1 -1];% dynamics determined by A
14 tau=0.01;% time discretization is tau
15 L=expm(A*tau);% forward semigroup operator
16 Sigma=sigma^2*((A+A')\ (L*L'-I));% dynamics noise variance integrated
17 sqrtS=sqrmt(Sigma);
18
19 m=zeros(N,J+1);v=m;z=zeros(J+1,1);c=zeros(N,N,J+1);% pre-allocate
20 v(:,1)=m0+sqrtm(C0)*randn(N,1);% initial truth
21 m(:,1)=10*randn(N,1);% initial mean/estimate
22 c(:,:,1)=100*C0;% initial covariance
23 z(1)=0;% initial ghost observation
24 H=[1,0];% observation operator
25
26 %% solution % assimilate!
27
28 for j=1:J
29     v(:,j+1)=L*v(:,j) + sqrtS*randn(N,1);% truth
30     z(j+1)=z(j)+tau*H*v(:,j+1) + gamma*sqrt(tau)*randn;% observation
31
32     mhat=L*m(:,j);% estimator intermediate "predict"
33     chat=L*c(:,:,j)*L'+Sigma;% covariance intermediate "predict"
34
35     d=(z(j+1)-z(j))/tau-H*mhat;% "innovation"
36     K=(tau*chat*H')/(H*chat*H'*tau+gamma^2);% "Kalman gain"
37     m(:,j+1)=mhat+K*d;% estimator "update"
38     c(:,:,j+1)=(I-K*H)*chat;% covariance "update"
39 end
40
41 figure;js=501;plot(tau*[0:js-1],v(2,1:js));hold;plot(tau*[0:js-1],
42     m(2,1:js)...%
43 , 'm'); plot(tau*[0:js-1],m(2,1:js)+reshape(sqrt(c(2,2,1:js)),1,js),'r--');
44 plot(tau*[0:js-1],m(2,1:js)-reshape(sqrt(c(2,2,1:js)),1,js),'r--');
45 hold;grid;xlabel('t');
46 title('Kalman-Bucy Filter');
47 figure;plot(tau*[0:J],reshape(c(1,1,:)+c(2,2,:),J+1,1));hold
48 plot(tau*[0:J],cumsum(reshape(c(1,1,:)+c(2,2,:),J+1,1))./[1:J+1]',...
49 'm','Linewidth',2); grid;hold;xlabel('t');axis([0 tau*J 0 50]);
50 title('Kalman-Bucy Filter Covariance');
51 figure;plot(tau*[0:J],sum((v-m).^2));hold;
52 plot(tau*[0:J],cumsum(sum((v-m).^2))./[1:J+1],'m','Linewidth',2);grid
53 hold;xlabel('t');axis([0 tau*J 0 50]);
54 title('Kalman-Bucy Filter Error')

```

9.3.2. p10c.m

Program p10c.m concerns the 3DVAR method and is similar to p10.m, using pseudodiscrete dynamics as a time-splitting method, with the forward model given by the Langevin equation with a double-well potential (9.1). The discrete-time Euler–Maruyama approximation of the continuous-time dynamics v and observation process z are defined in lines 24–25, as in the previous sections. The time step τ is set in line 18. Recall that the dynamics of $C = C_0$ are trivial for 3DVAR, and the update in line 28 is the approximation of the continuous 3DVAR filter over time step τ :

$$\frac{dm}{dt} = f(m) + CH^T \Gamma_0^{-1} \left(\frac{dz}{dt} - Hm \right), \quad m(0) = m_0.$$

The results are presented in Figs. 8.2 and 8.3.

```

1 clear;set(0,'defaultaxesfontsize',20);format long
2 %% p10c.m Continuous 3DVAR Filter, double-well
3 %% setup
4
5 J=1e4;% number of steps
6 alpha=2.5;% dynamics determined by alpha
7 gamma=1e-1;% observational noise variance is gamma^2
8 sigma=3e-1;% dynamics noise variance is sigma^2
9 C0=1;% prior initial condition variance
10 m0=0;% prior initial condition mean
11 sd=1;rng(sd);% choose random number seed
12
13 m=zeros(J,1);v=m;z=m;z(1)=0;% pre-allocate
14 v(1)=m0+sqrt(C0)*randn;% initial truth
15 m(1)=2*randn;% initial mean/estimate
16 eta=.1;% stabilization coefficient 0 < eta << 1
17 c=gamma^2/eta;H=1;% covariance and observation operator
18 tau=0.01;% time discretization is tau
19 K=tau*(c*H')/(H*c*H'*tau+gamma^2);% Kalman gain
20
21 %% solution % assimilate!
22
23 for j=1:J
24     v(j+1)=v(j)+tau*alpha*(v(j)-v(j)^3) + sigma*sqrt(tau)*randn;% truth
25     z(j+1)=z(j)+tau*H*v(j+1) + gamma*sqrt(tau)*randn;% observation
26     mhat=m(j)+tau*alpha*(m(j)-m(j)^3);% estimator predict
27     d=(z(j+1)-z(j))/tau-H*mhat;% innovation
28     m(j+1)=mhat+K*d;% estimator update
29 end
30 js=201;% plot truth, mean, standard deviation, observations
31 figure;plot(tau*[0:js-1],v(1:js));hold;plot(tau*[0:js-1],m(1:js),'m');
32 plot(tau*[0:js-1],m(1:js)+sqrt(c),'r--',tau*[0:js-1],m(1:js)-sqrt(c),
33      'r--');
34 hold;grid;xlabel('t');title('3DVAR Filter')
35
36 figure;plot(tau*[0:J],c*[0:J].^0);hold
37 plot(tau*[0:J],c*[0:J].^0,'m','Linewidth',2);grid
38 hold;xlabel('t');title('3DVAR Filter Covariance');
39
40 figure;plot(tau*[0:J],(v-m).^2);hold;
41 plot(tau*[0:J],cumsum((v-m).^2)/[1:J+1]', 'm','Linewidth',2);grid
42 hold;xlabel('t');title('3DVAR Filter Error')
```

9.3.3. p11c.m

Program `p11c.m` is a modification of `p11.m` using the similar split-step approximations described in Sections 9.3.1 and 9.3.2 for the continuous-time extended Kalman filter. The forward model is again the Langevin equation with a double-well potential (9.1). The updates in lines 32–33 this time are approximations of

$$\begin{aligned}\frac{dm}{dt} &= f(m) + CH^T \Gamma_0^{-1} \left(\frac{dz}{dt} - Hm \right), \quad m(0) = m_0, \\ \frac{dC}{dt} &= Df(m)C + C(Df(m))^T + \Sigma_0 - CH^T \Gamma_0^{-1} HC, \quad C(0) = C_0.\end{aligned}$$

Notice that the split-step scheme preserves the symmetry and positivity of C thanks to the approximations used in lines 27–28 and 33. The results are presented in Fig. 8.4.

9.3.4. p12c.m

Program `p12c.m` is a modification of `p12.m` using the similar split-step approximations described in Sections 9.3.1 and 9.3.2 for the continuous-time EnKF. The forward model is again the Langevin equation with a double-well potential (9.1). The updates in lines 34, 32, and 26 are given by

$$\frac{dv^{(n)}}{dt} = f(v^{(n)}) + C(v)H^T \Gamma_0^{-1} \left(\frac{dz^{(n)}}{dt} - Hv^{(n)} \right) + \Sigma_0^{1/2} dB_v, \quad (9.7a)$$

$$\frac{dz^{(n)}}{dt} = Hv + \Gamma_0^{1/2} \left(\frac{dW^{(n)}}{dt} + \frac{dB_z}{dt} \right). \quad (9.7b)$$

Notice that the increments $dW^{(n)}$ are added in line 32, expanding the single observation from line 26 into an ensemble. The covariance is updated in line 36:

$$\begin{aligned}m &= \frac{1}{N} \sum_{n=1}^N v^{(n)}, \\ C(v) &= \frac{1}{N-1} \sum_{n=1}^N (v^{(n)} - m)(v^{(n)} - m)^T.\end{aligned}$$

The results are presented in Fig. 8.5.

```

1 clear;set(0,'defaultaxesfontsize',20);format long
2 %% p11c.m Extended Kalman-Bucy Filter, double-well
3 %% setup
4
5 J=1e4;% number of steps
6 alpha=2.5;% dynamics determined by alpha
7 gamma=.1;% observational noise variance is gamma^2
8 sigma=3e-1;% dynamics noise variance is sigma^2
9 C0=1;% prior initial condition variance
10 m0=0;% prior initial condition mean
11 sd=1;rng(sd);% choose random number seed
12
13 m=zeros(J,1);v=m;z=m;z(1)=0;c=m;% pre-allocate
14 v(1)=m0+sqrt(C0)*randn;% initial truth
15 m(1)=2*randn;% initial mean/estimate
16 c(1)=10*C0;H=1;% initial covariance and observation operator
17 tau=0.01;% time discretization is tau
18
19 %% solution % assimilate!
20
21 for j=1:J
22
23     v(j+1)=v(j)+tau*alpha*(v(j)-v(j)^3) + sigma*sqrt(tau)*randn;% truth
24     z(j+1)=z(j)+tau*H*v(j+1) + gamma*sqrt(tau)*randn;% observation
25
26     mhat=m(j)+tau*alpha*(m(j)-m(j)^3);% estimator predict
27     chat=(1+tau*alpha*(1-3*m(j)^2))*c(j)*...
28         (1+tau*alpha*(1-3*m(j)^2))+sigma^2*tau;% covariance predict
29
30     d=(z(j+1)-z(j))/tau-H*mhat;% innovation
31     K=(tau*chat*H')/(H*chat*H'*tau+gamma^2);% Kalman gain
32     m(j+1)=mhat+K*d;% estimator update
33     c(j+1)=(1-K*H)*chat;% covariance update
34
35 end
36
37 js=201;% plot truth, mean, standard deviation, observations
38 figure;plot(tau*[0:js-1],v(1:js));hold;plot(tau*[0:js-1],m(1:js),'m');
39 plot(tau*[0:js-1],m(1:js)+sqrt(c(1:js)),'r--');
40 plot(tau*[0:js-1],m(1:js)-sqrt(c(1:js)),'r--');hold;grid;xlabel('t');
41 title('ExKF')
42
43 figure;plot(tau*[0:J],c);hold
44 plot(tau*[0:J],cumsum(c)./[1:J+1]','m','Linewidth',2);grid
45 hold;xlabel('t');
46 title('ExKF Covariance');
47
48 figure;plot(tau*[0:J],(v-m).^2);hold;
49 plot(tau*[0:J],cumsum((v-m).^2)./[1:J+1]','m','Linewidth',2);grid
50 hold;xlabel('t');
51 title('ExKF Error')

```

```

1 clear;set(0,'defaultaxesfontsize',20);format long
2 %% p12c.m Ensemble Kalman Filter (PO), double-well
3 %% setup
4
5 J=1e4;% number of steps
6 alpha=2.5;% dynamics determined by alpha
7 gamma=.1;% observational noise variance is gamma^2
8 sigma=3e-1;% dynamics noise variance is sigma^2
9 C0=1;% prior initial condition variance
10 m0=0;% prior initial condition mean
11 sd=1;rng(sd);% choose random number seed
12 N=10;% number of ensemble members
13
14 m=zeros(J,1);v=m;z=m;z(1)=0;c=m;U=zeros(J,N);% pre-allocate
15 v(1)=m0+sqrt(C0)*randn;% initial truth
16 m(1)=2*randn;% initial mean/estimate
17 c(1)=10*C0;H=1;% initial covariance and observation operator
18 U(1,:)=m(1)+sqrt(c(1))*randn(1,N);m(1)=sum(U(1,:))/N;% initial ensemble
19 tau=0.01;st=sigma*sqrt(tau);% time discretization is tau
20
21 %% solution % assimilate!
22
23 for j=1:J
24
25     v(j+1)=v(j)+tau*alpha*(v(j)-v(j)^3) + st*randn;% truth
26     z(j+1)=z(j)+tau*H*v(j+1) + gamma*sqrt(tau)*randn;% observation
27
28     Uhat=U(j,:)+tau*alpha*(U(j,:)-U(j,:).^3)+st*randn(1,N);% ensemble
29         predict
30     mhat=sum(Uhat)/N;% estimator predict
31     chat=(Uhat-mhat)*(Uhat-mhat)'/(N-1);% covariance predict
32
33     d=(z(j+1)-z(j)+sqrt(tau)*gamma*randn(1,N))/tau-H*Uhat;% innovation
34     K=(tau*chat*H')/(H*chat*H'*tau+gamma^2);% Kalman gain
35     U(j+1,:)=Uhat+K*d;% ensemble update
36     m(j+1)=sum(U(j+1,:))/N;% estimator update
37     c(j+1)=(U(j+1,:)-m(j+1))*(U(j+1,:)-m(j+1))'/(N-1);% covariance update
38
39 end
40
41 js=201;% plot truth, mean, standard deviation, observations
42 figure(1);plot(tau*[0:js-1],v(1:js));hold;plot(tau*[0:js-1],m(1:js),'m');
43 plot(tau*[0:js-1],m(1:js)+sqrt(c(1:js)),r--');
44 plot(tau*[0:js-1],m(1:js)-sqrt(c(1:js)),r--');hold;grid;xlabel('t');
45 title('EnKF')
46
47 figure(2);plot(tau*[0:J],c);hold
48 plot(tau*[0:J],cumsum(c)./[1:J+1]',m', 'Linewidth',2);grid
49 hold;xlabel('t');title('EnKF Covariance');
50
51 figure(3);plot(tau*[0:J],(v-m).^2);hold;
52 plot(tau*[0:J],cumsum((v-m).^2)./[1:J+1]',m', 'Linewidth',2);grid
53 hold;xlabel('t');title('EnKF Error')

```

9.3.5. p13c.m

Program `p13c.m` is a modification of `p13.m` using the similar split-step approximations described in Sections 9.3.1 and 9.3.2 for the continuous-time ensemble transform version of the EnKF—the ETKF. The forward model is again the Langevin equation with a double-well potential (9.1). This program is not studied in Chapter 8. The results are presented in Fig. 8.6.

9.3.6. p14c.m

Program `p14c.m` is a modification of `p14.m` using the similar split-step approximations described in Sections 9.3.1 and 9.3.2 for the continuous-time particle filter with standard proposal. The forward model is again the Langevin equation with a double-well potential (9.1). This program is not studied in Chapter 8, although it follows from the discussion in Section 8.3. The results are presented in Fig. 8.7.

9.3.7. p15c.m

Program `p15c.m` is a modification of `p15.m` using the similar split-step approximations described in Sections 9.3.1 and 9.3.2 for the continuous-time particle filter with optimal proposal. The forward model is again the Langevin equation with a double-well potential (9.1). This program is not studied in Chapter 8, although it follows from the discussion in Section 8.3. The results are presented in Fig. 8.8

```

1 clear;set(0,'defaultaxesfontsize',20);format long
2 %% p13c.m Ensemble Kalman Filter (ETKF), double-well
3 %% setup
4
5 J=1e4;% number of steps
6 alpha=2.5;% dynamics determined by alpha
7 gamma=.1;% observational noise variance is gamma^2
8 sigma=3e-1;% dynamics noise variance is sigma^2
9 C0=1;% prior initial condition variance
10 m0=0;% prior initial condition mean
11 sd=1;rng(sd);% choose random number seed
12 N=10;% number of ensemble members
13
14 m=zeros(J,1);v=m;z=m;z(1)=0;c=m;U=zeros(J,N);% pre-allocate
15 v(1)=m0+sqrt(C0)*randn;% initial truth
16 m(1)=2*randn;% initial mean/estimate
17 c(1)=10*C0;H=1;% initial covariance and observation operator
18 U(1,:)=m(1)+sqrt(c(1))*randn(1,N);m(1)=sum(U(1,:))/N;% initial ensemble
19 tau=0.01;st=sigma*sqrt(tau);% time discretization is tau
20
21 %% solution % assimilate!
22
23 for j=1:J
24
25     v(j+1)=v(j)+tau*alpha*(v(j)-v(j)^3) + st*randn;% truth
26     z(j+1)=z(j)+tau*H*v(j+1) + gamma*sqrt(tau)*randn;% observation
27
28     Uhat=U(j,:)+tau*alpha*(U(j,:)-U(j,:).^3)+st*randn(1,N);% ensemble
29         predict
30     mhat=sum(Uhat)/N;% estimator predict
31     Xhat=(Uhat-mhat)/sqrt(N-1);% centered ensemble
32     chat=Xhat*Xhat';% covariance predict
33     T=sqrtn(inv(eye(N)+Xhat'*H'*H*Xhat*tau/gamma^2));% square-root
34         transform
35     X=Xhat*T;% transformed centered ensemble
36
37     d=(z(j+1)-z(j))/tau-H*mhat;randn(1,N);% innovation
38     K=(tau*chat*H')/(H*chat*H'*tau+gamma^2);% Kalman gain
39     m(j+1)=mhat+K*d;% estimator update
40     U(j+1,:)=m(j+1)+X*sqrt(N-1);% ensemble update
41     c(j+1)=X*X';% covariance update
42
43 end
44 js=201;% plot truth, mean, standard deviation, observations
45 figure;plot(tau*[0:js-1],v(1:js));hold;plot(tau*[0:js-1],m(1:js),'m');
46 plot(tau*[0:js-1],m(1:js)+sqrt(c(1:js)),'r--');
47 plot(tau*[0:js-1],m(1:js)-sqrt(c(1:js)),'r--');hold;grid;xlabel('t');
48 title('EnKF(ETKF)');
49 figure;plot(tau*[0:J],c);hold
50 plot(tau*[0:J],cumsum(c)./[1:J+1]','m','Linewidth',2);grid
51 hold;xlabel('t');title('EnKF(ETKF) Covariance');
52 figure;plot(tau*[0:J],(v-m).^2);hold;
53 plot(tau*[0:J],cumsum((v-m).^2)./[1:J+1]','m','Linewidth',2);grid
54 hold;xlabel('t');title('EnKF(ETKF) Error')

```

```

1 clear;set(0,'defaultaxesfontsize',20);format long
2 %% p14c.m Particle Filter (SIRS), double-well
3 %% setup
4
5 J=1e4;% number of steps
6 alpha=2.5;% dynamics determined by alpha
7 gamma=.1;% observational noise variance is gamma^2
8 sigma=3e-1;% dynamics noise variance is sigma^2
9 C0=1;% prior initial condition variance
10 m0=0;% prior initial condition mean
11 sd=1;rng(sd);% Choose random number seed
12 N=10;% number of ensemble members
13
14 m=zeros(J,1);v=m;z=m;z(1)=0;c=m;U=zeros(J,N);% pre-allocate
15 v(1)=m0+sqrt(C0)*randn;% initial truth
16 m(1)=2*randn;% initial mean/estimate
17 c(1)=10*C0;H=1;% initial covariance and observation operator
18 U(1,:)=m(1)+sqrt(c(1))*randn(1,N);m(1)=sum(U(1,:))/N;% initial ensemble
19 tau=0.01;st=sigma*sqrt(tau);% time discretization is tau
20
21 %% solution % Assimilate!
22
23 for j=1:J
24
25     v(j+1)=v(j)+tau*alpha*(v(j)-v(j)^3) + st*randn;% truth
26     z(j+1)=z(j)+tau*H*v(j+1) + gamma*sqrt(tau)*randn;% observation
27
28     Uhat=U(j,:)+tau*alpha*(U(j,:)-U(j,:).^3)+st*randn(1,N);%ensemble
29         predict
30     d=(z(j+1)-z(j))/tau-H*Uhat;% ensemble innovation
31     what=exp(-1/2*(tau/gamma^2*d.^2));% weight update
32     w=what/sum(what); randn(1,N);% normalize predict weights
33
34     ws=cumsum(w);% resample: compute cdf of weights
35     for n=1:N
36         ix=find(ws>rand,1,'first');% resample (i)
37         U(j+1,n)=Uhat(ix);% resample (ii)
38     end
39
40     m(j+1)=sum(U(j+1,:))/N;% estimator update
41     c(j+1)=(U(j+1,:)-m(j+1))*(U(j+1,:)-m(j+1))'/N;% covariance update
42
43 end
44 js=201;% plot truth, mean, standard deviation, observations
45 figure(1);plot(tau*[0:js-1],v(1:js));hold;plot(tau*[0:js-1],m(1:js),'m');
46 plot(tau*[0:js-1],m(1:js)+sqrt(c(1:js)),'r--');
47 plot(tau*[0:js-1],m(1:js)-sqrt(c(1:js)),'r--');hold;grid;xlabel('t');
48 title('Particle Filter (Standard)');
49 figure(2);plot(tau*[0:J],c);hold
50 plot(tau*[0:J],cumsum(c)./[1:J+1]','m','Linewidth',2);grid
51 hold;xlabel('t');title('Particle Filter (Standard) Covariance');
52 figure(3);plot(tau*[0:J],(v-m).^2);hold;
53 plot(tau*[0:J],cumsum((v-m).^2)./[1:J+1]','m','Linewidth',2);grid
54 hold;xlabel('t');title('Particle Filter (Standard) Error')

```

```

1 clear;set(0,'defaultaxesfontsize',20);format long
2 %% p15c.m Particle Filter (SIRS, OP), double-well
3 %% setup
4 J=1e4;% number of steps
5 alpha=2.5;% dynamics determined by alpha
6 gamma=.1;% observational noise variance is gamma^2
7 sigma=3e-1;% dynamics noise variance is sigma^2
8 m0=0; C0=1;% prior initial condition mean and variance
9 sd=1;rng(sd);% Choose random number seed
10 N=10;% number of ensemble members
11
12 m=zeros(J,1);v=m;z=m;z(1)=0;c=m;U=zeros(J,N);% pre-allocate
13 v(1)=m0+sqrt(C0)*randn;% initial truth
14 m(1)=2*randn;% initial mean/estimate
15 c(1)=10*C0;H=1;% initial covariance and observation operator
16 U(1,:)=m(1)+sqrt(c(1))*randn(1,N);m(1)=sum(U(1,:))/N;% initial ensemble
17 tau=0.01;st=sigma*sqrt(tau);% time discretization is tau
18
19 %% solution % Assimilate!
20 for j=1:J
21     v(j+1)=v(j)+tau*alpha*(v(j)-v(j)^3) + st*randn;% truth
22     z(j+1)=z(j)+tau*H*v(j+1) + gamma*sqrt(tau)*randn;% observation
23
24     Sig=inv(inv(sigma^2*tau)+H'*inv(gamma^2/tau)*H);% optimal proposal cov
25     em=Sig*(inv(sigma^2*tau)*(U(j,:)+tau*alpha*(U(j,:)-U(j,:).^3))+...
26             H'*inv(gamma^2/tau)*(z(j+1)-z(j))/tau);% optimal proposal mean
27     Uhat=em+sqrt(Sig)*randn(1,N);% ensemble optimally importance sampled
28
29     d=(z(j+1)-z(j))/tau-H*(U(j,:)+tau*alpha*(U(j,:)-U(j,:).^3));% ensemble
30         innov
31     what=exp(-1/2/(sigma^2*tau+gamma^2/tau)*d.^2);% weight update
32     w=what/sum(what);randn(1,N);% normalize predict weights
33
34     ws=cumsum(w);% resample: compute cdf of weights
35     for n=1:N
36         ix=find(ws>rand,1,'first');% resample (i)
37         U(j+1,n)=Uhat(ix);% resample (ii)
38     end
39
40     m(j+1)=sum(U(j+1,:))/N;% estimator update
41     c(j+1)=(U(j+1,:)-m(j+1))*(U(j+1,:)-m(j+1))'/N;% covariance update
42 end
43
44 js=201;% plot truth, mean, standard deviation, observations
45 figure(1);plot(tau*[0:js-1],v(1:js));hold;plot(tau*[0:js-1],m(1:js),'m');
46 plot(tau*[0:js-1],m(1:js)+sqrt(c(1:js)), 'r--');
47 plot(tau*[0:js-1],m(1:js)-sqrt(c(1:js)), 'r--');hold;grid;xlabel('t');
48 title('Particle Filter (Optimal)');
49 figure(2);plot(tau*[0:J],(v-m).^2);hold;
50 plot(tau*[0:J],cumsum((v-m).^2)./[1:J+1]', 'm', 'Linewidth',2);grid
51 hold;xlabel('t');title('Particle Filter (Optimal) Error')
52 figure(3);plot(tau*[0:J],(v-m).^2);hold;
53 plot(tau*[0:J],cumsum((v-m).^2)./[1:J+1]', 'm', 'Linewidth',2);grid
54 hold;xlabel('t');
55 title('Particle Filter (Optimal) Error')

```

9.3.8. p16c.m

Program `p16c.m` is an implementation of the extended Kalman filter for the Lorenz '63 model (6.23). As in `p7c.m`, this program is written as a function rather than a script to facilitate the use of the auxiliary function `f` defined on line 44. The forward solver is now given by a simple Euler–Maruyama discretization, on line 19. An additional auxiliary function `Df` is defined to evaluate the derivative matrix of the right-hand side for a given value of state, in this case $m(t_j = j\tau)$. The rest of the program proceeds as in the program `11c.m`. The output of this program with line 18 commented and 19 uncommented is presented in Figure 8.9, and the output with line 18 uncommented and 19 commented is presented in Figure 8.10, using plotting commands similar to those on lines 35–42.

9.3.9. p17c.m

Similarly to `p16c.m`, the program `p17c.m` is an implementation of the extended Kalman filter for the Lorenz '96 model (6.24), written as a function. Again auxiliary functions are defined for the right-hand side on line 44 and the derivative matrix on line 47. The rest of the program is also very similar to `p16c.m`, except now one is free to choose the dimension, defined by $N = r*n$ on line 6. Thus one can easily choose a fraction q/r of the components to observe by letting the observation matrix H be block diagonal with $n q \times r$ blocks, each consisting of a q -dimensional identity block next to a $(q \times q-r)$ -dimensional zero block. This $(q*n \times N)$ -dimensional matrix is constructed on lines 18–19. If one changes the number of steps to $J=2e5$ on line 7, then one obtains as output Figure 8.11. If one then modifies line 6 so that $q=1$, then one obtains the output of Figure 8.12.

```

1 function this=p16c
2 clear;set(0,'defaultaxesfontsize',20);format long
3 %% p16c.m ExKF, Lorenz 63
4 %% setup
5
6 J=1e5;% number of steps
7 a=10;b=8/3;r=28;% define parameters
8 gamma=2e-1;% observational noise variance is gamma^2
9 sigma=2e0;% dynamics noise variance is sigma^2
10 I=eye(3);C0=I;% prior initial condition covariance
11 m0=zeros(3,1);% prior initial condition mean
12 sd=1;rng(sd);% choose random number seed
13
14 m=zeros(3,J);v=m;z=m(:,1:);z(1)=0;c=zeros(3,3,J);% pre-allocate
15 v(:,1)=m0+sqrtm(C0)*randn(3,1);% initial truth
16 m(:,1)=10*randn(3,1);% initial mean/estimate
17 c(:,:,1)=10*C0;% initial covariance operator
18 H=[0,0,1];% observation operator
19 %H=[1,0,0];% observation operator
20 tau=1e-4;% time discretization is tau
21
22 %% solution % assimilate!
23 for j=1:J
24     v(:,j+1)=v(:,j)+tau*f(v(:,j),a,b,r) + sigma*sqrt(tau)*randn(3,1);%
25         truth
26     z(:,j+1)=z(:,j)+tau*H*v(:,j+1) + gamma*sqrt(tau)*randn;% observation
27     mhat=m(:,j)+tau*f(m(:,j),a,b,r);% estimator predict
28     chat=(I+tau*Df(m(:,j),a,b))*c(:,:,j)*...
29             (I+tau*Df(m(:,j),a,b))'+sigma^2*tau*I;% covariance predict
30     d=(z(j+1)-z(j))/tau-H*mhat;% innovation
31     K=(tau*chat*H')/(H*chat*H'*tau+gamma^2);% Kalman gain
32     m(:,j+1)=mhat+K*d;% estimator update
33     c(:,:,j+1)=(I-K*H)*chat;% covariance update
34 end
35
36 figure;js=j;
37 plot(tau*[0:js-1],v(2,1:js));hold;plot(tau*[0:js-1],m(2,1:js),'m');
38 plot(tau*[0:js-1],m(2,1:js)+reshape(sqrt(c(2,2,1:js)),1,js),'r--');
39 plot(tau*[0:js-1],m(2,1:js)-reshape(sqrt(c(2,2,1:js)),1,js),'r--');
40 hold;grid;xlabel('t');legend('u_2','m_2','m_2 \pm c_2^{1/2}');
41 title('ExKF, L63, u_1 observed');Jj=min(J,j);
42 figure;plot(tau*[0:Jj-1],sum((v(:,1:Jj)-m(:,1:Jj)).^2));hold;
43 grid;hold;xlabel('t');title('ExKF, L63, MSE, u_1 observed')
44
45 function rhs=f(y,a,b,r)
46 rhs(1,1)=a*(y(2)-y(1));
47 rhs(2,1)=-a*y(1)-y(2)-y(1)*y(3);
48 rhs(3,1)=y(1)*y(2)-b*y(3)-b*(r+a);
49 function A=Df(y,a,b)
50 A(1,:)=[-a,a,0];
51 A(2,:)=[-a-y(3),-1,-y(1)];
52 A(3,:)=[y(2),y(1),-b];

```

```

1 function this=p17c
2 clear;set(0,'defaultaxesfontsize',20);format long
3 %% p17c.m ExKF, Lorenz 96
4 %% setup
5
6 q=2;r=3;n=2;N=r*n;% observe q/r coordinates in N dimensions
7 J=5e4;F=8,% number of steps and parameter
8 gamma=1e-1;% observational noise variance is gamma^2
9 sigma=1e0;% dynamics noise variance is sigma^2
10 I=eye(N);C0=I;% prior initial condition covariance
11 m0=zeros(N,1);% prior initial condition mean
12 sd=1;rng(sd);% choose random number seed
13
14 m=zeros(N,J);v=m(1:q*n,:);c=zeros(N,N,J);% pre-allocate
15 v(:,1)=m0+sqrtm(C0)*randn(N,1);% initial truth
16 m(:,1)=10*randn(N,1);% initial mean/estimate
17 c(:,:,1)=25*C0;% initial covariance operator
18 H=zeros(q*n,N);for k=1:n;H(q*(k-1)+1:q*k,r*(k-1)+1:r*k)= ...
19 [eye(q),zeros(q,r-q)];end% observation operator
20 tau=1e-4;% time discretization is tau
21
22 %% solution % assimilate!
23 for j=1:J
24     v(:,j+1)=v(:,j)+tau*f(v(:,j),F) + sigma*sqrt(tau)*randn(N,1);% truth
25     z(:,j+1)=z(:,j)+tau*H*v(:,j+1) + gamma*sqrt(tau)*randn(q*n,1);%
26         observation
27     mhat=m(:,j)+tau*f(m(:,j),F);% estimator predict
28     chat=(I+tau*Df(m(:,j),N))*c(:,:,j)* ...
29         (I+tau*Df(m(:,j),N))'+sigma^2*tau*I;% covariance predict
30     d=(z(:,j+1)-z(:,j))/tau-H*mhat;% innovation
31     K=(tau*chat*H')/(H*chat*H'*tau+gamma^2*eye(q*n));% Kalman gain
32     m(:,j+1)=mhat+K*d;% estimator update
33     c(:,:,j+1)=(I-K*H)*chat;% covariance update
34 end
35
36 figure;js=j;
37 plot(tau*[0:js-1],v(2,1:js));hold;plot(tau*[0:js-1],m(2,1:js),'m');
38 plot(tau*[0:js-1],m(2,1:js)+reshape(sqrt(c(2,2,1:js)),1,js),'r--');
39 plot(tau*[0:js-1],m(2,1:js)-reshape(sqrt(c(2,2,1:js)),1,js),'r--');
40 hold;grid;xlabel('t');legend('u_2','m_2','m_2 \pm c_2^{1/2}');
41 title('ExKF, L96, 2/3 observed');Jj=min(J,j);
42 figure;plot(tau*[0:Jj-1],sum((v(:,1:Jj)-m(:,1:Jj)).^2/N));hold;
43 grid;hold;xlabel('t');title('ExKF, L96, MSE, 2/3 observed')
44
45 function rhs=f(y,F)
46 rhs=[y(end);y(1:end-1)].*[y(2:end);y(1)] - ...
47 [y(end-1:end);y(1:end-2)]) - y + F*y.^0;
48 function A=Df(y,N)
49 A=-eye(N);
50 A=A+diag([y(end);y(1:end-2)],1);A(end,1)=y(end-1);
51 A=A+diag(y(2:end-1),-2);A(1,end-1)=y(end);A(2,end)=y(1);
52 A=A+diag([y(3:end);y(1)] - [y(end);y(1:end-2)]),-1);
53 A(1,end)=y(2)-y(end-1);

```

References

1. H. Abarbanel. *Predicting the Future: Completing Models of Observed Complex Systems*. Springer, 2013.
2. J. Anderson. A method for producing and evaluating probabilistic forecasts from ensemble model integrations. *Journal of Climate*, 9(7):1518–1530, 1996.
3. J. Anderson. An ensemble adjustment Kalman filter for data assimilation. *Mon. Wea. Rev.*, 129(12):2884–2903, 2001.
4. J. Anderson and S. Anderson. A Monte Carlo implementation of the nonlinear filtering problem to produce ensemble assimilations and forecasts. *Mon. Wea. Rev.*, 127(12):2741–2758, 1999.
5. A. Andrews. A square root formulation of the Kalman covariance equations. *AIAA Journal*, 6(6):1165–1166, 1968.
6. A. Apte, C. K. R. T. Jones, A. M. Stuart, and J. Voss. Data assimilation: mathematical and statistical perspectives. *Int. J. Num. Meth. Fluids*, 56:1033–1046, 2008.
7. L. Arnold. *Random Dynamical Systems*. Springer Monographs in Mathematics. Springer-Verlag, Berlin, 1998.
8. A. Azouani, E. Olson, and E. Titi. Continuous data assimilation using general interpolant observables. *Journal of Nonlinear Science*, pages 1–28, 2013.
9. A. Bain and D. Crisan. *Fundamentals of Stochastic Filtering*, volume 3. Springer, 2009.
10. V. Baladi. *Positive Transfer Operators and Decay of Correlations*. World Scientific, 2000.
11. T. Bengtsson, P. Bickel, and B. Li. Curse of dimensionality revisited: the collapse of importance sampling in very large scale systems. *IMS Collections: Probability and Statistics: Essays in Honor of David Freedman*, 2:316–334, 2008.
12. A. Bennett. *Inverse Modeling of the Ocean and Atmosphere*. Cambridge, 2002.
13. J. O. Berger. *Statistical Decision Theory and Bayesian Analysis*. Springer Verlag, 1985.
14. N. Berglund and B. Gentz. *Noise-Induced Phenomena in Slow-fast Dynamical Systems*. Probability and its Applications (New York). Springer-Verlag London Ltd., London, 2006. A sample-paths approach.
15. J. Bernardo and A. Smith. *Bayesian Theory*. Wiley, 1994.
16. A. Beskos, D. Crisan, A. Jasra, et al. On the stability of sequential Monte Carlo methods in high dimensions. *The Annals of Applied Probability*, 24(4):1396–1445, 2014.
17. A. Beskos, O. Papaspiliopoulos, G. O. Roberts, and P. Fearnhead. Exact and computationally efficient likelihood-based estimation for discretely observed diffusion processes (with discussion). *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68(3):333–382, 2006.
18. A. Beskos, G. O. Roberts, A. M. Stuart, and J. Voss. MCMC methods for diffusion bridges. *Stochastic Dynamics*, 8(3):319–350, Sep 2008.
19. P. Bickel, B. Li, and T. Bengtsson. Sharp failure rates for the bootstrap particle filter in high dimensions. *IMS Collections: Pushing the Limits of Contemporary Statistics*, 3:318–329, 2008.
20. C. Bishop, B. Etherton, and S. Majumdar. Adaptive sampling with the ensemble transform Kalman filter. Part I: Theoretical aspects. *Mon. Wea. Rev.*, 129(3):420–436, 2001.
21. D. Blömker, K. J. H. Law, A. M. Stuart, and K. C. Zygalakis. Accuracy and stability of the continuous-time 3DVAR filter for the Navier-Stokes equation. *Nonlinearity*, 26(8):2193, 2013.
22. M. Branicki and A. Majda. Quantifying Bayesian filter performance for turbulent dynamical systems through information theory. *Comm. Math. Sci*, 2014.
23. L. Breiman. Probability, volume 7 of classics in applied mathematics. *Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA*, 1992.
24. C. E. A. Brett, K. F. Lam, K. J. H. Law, D. S. McCormick, M. R. Scott, and A. M. Stuart. Accuracy and stability of filters for dissipative pdes. *Physica D: Nonlinear Phenomena*, 245(1):34–45, 2013.

25. S. Brooks and A. Gelman. General methods for monitoring convergence of iterative simulations. *Journal of Computational and Graphical Statistics*, 7(4):434–455, 1998.
26. A. Bryson and M. Frazier. Smoothing for linear and nonlinear dynamic systems. In *Proceedings Optimum System Synthesis Conference*. US Air Force Tech. Rep. AFB-TDR-63-119, 1963.
27. A. Carrassi, M. Ghil, A. Trevisan, and F. Ubaldi. Data assimilation as a nonlinear dynamical systems problem: Stability and convergence of the prediction-assimilation system. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 18:023112, 2008.
28. A. Chorin, M. Morzfeld, and X. Tu. Implicit particle filters for data assimilation. *Comm. Appl. Math. Comp. Sc.*, 5:221–240, 2010.
29. A. Chorin and X. Tu. Interpolation and iteration for nonlinear filters. <http://arxiv.org/abs/0910.3241>.
30. A. Chorin and X. Tu. Implicit sampling for particle filters. *Proc. Nat. Acad. Sc.*, 106:17249–17254, 2009.
31. C. Cotter and S. Reich. *Probabilistic Forecasting and Bayesian Data Assimilation*. Cambridge University Press, 2015.
32. S. L. Cotter, M. Dashti, J. C. Robinson, and A. M. Stuart. Bayesian inverse problems for functions and applications to fluid mechanics. *Inverse Problems*, 25(11):115008, 2009.
33. S. L. Cotter, M. Dashti, and A. M. Stuart. Approximation of Bayesian inverse problems for PDEs. *SIAM J. Num. Anal.*, 48(1):322–345, 2010.
34. S. L. Cotter, M. Dashti, and A. M. Stuart. Variational data assimilation using targeted random walks. *Int. J. Num. Meth. Fluids*, 2011.
35. S. L. Cotter, G. Roberts, A. M. Stuart, and D. White. MCMC methods for functions: modifying old algorithms to make them faster. *Statistical Science*, 28:424–446, 2013.
36. P. Courtier, E. Andersson, W. Heckley, D. Vasiljevic, M. Hamrud, A. Hollingsworth, F. Rabier, M. Fisher, and J. Pailleux. The ECMWF implementation of three-dimensional variational assimilation (3d-Var). I: Formulation. *Quart. J. R. Met. Soc.*, 124(550):1783–1807, 1998.
37. P. Courtier and O. Talagrand. Variational assimilation of meteorological observations with the adjoint vorticity equation. II: numerical results. *Quart. J. Royal Met. Soc.*, 113:1329–1347, 1987.
38. D. Crisan, J. Diehl, P. Friz, and H. Oberhauser. Robust filtering: Correlated noise and multidimensional observation. *The Annals of Applied Probability*, 23(5):2139–2160, 2013.
39. D. Crisan and A. Doucet. A survey of convergence results on particle filtering methods for practitioners. *Signal Processing, IEEE Transactions on*, 50(3):736–746, 2002.
40. B. Dacarogna. *Direct Methods in the Calculus of Variations*. Springer, New York, 1989.
41. G. DaPrato and J. Zabczyk. *Stochastic Equations in Infinite Dimensions*, volume 44 of *Encyclopedia of Mathematics and Its Applications*. Cambridge University Press, Cambridge, 1992.
42. M. Dashti, S. Harris, and A. M. Stuart. Besov priors for Bayesian inverse problems. *Inverse Problems and Imaging*, 6:183–200, 2012.
43. M. Dashti, K. J. H. Law, A. M. Stuart, and J. Voss. MAP estimators and posterior consistency in Bayesian nonparametric inverse problems. *Inverse Problems*, 29:095017, 2013.
44. M. Dashti and A. M. Stuart. Uncertainty quantification and weak approximation of an elliptic inverse problem. *SIAM J. Num. Anal.*, 49:2524–2542, 2011.
45. P. Del Moral. *Feynman-Kac Formulae*. Springer, 2004.
46. P. Del Moral and A. Guionnet. On the stability of interacting processes with applications to filtering and genetic algorithms. In *Annales de l'Institut Henri Poincaré (B) Probability and Statistics*, volume 37, pages 155–194. Elsevier, 2001.
47. A. Doucet, S. Godsill, and C. Andrieu. On sequential Monte Carlo sampling methods for Bayesian filtering. *Statistics and computing*, 10(3):197–208, 2000.
48. N. Doucet, A. de Freitas and N. Gordon. *Sequential Monte Carlo in Practice*. Springer-Verlag, 2001.
49. K. D. Elworthy. *Stochastic Differential Equations on Manifolds*, volume 70 of *London Mathematical Society Lecture Note Series*. Cambridge University Press, 1982.
50. G. Evensen. *Data Assimilation: The Ensemble Kalman Filter*. Springer, 2006.
51. K. J. Falconer. *The Geometry of Fractal Sets*, volume 85. Cambridge University Press, 1986.
52. M. Fisher, Mand Leutbecher and G. Kelly. On the equivalence between Kalman smoothing and weak-constraint four-dimensional variational data assimilation. *Q. J. Roy. Met. Soc.*, 131(613):3235–3246, 2005.
53. M. I. Freidlin and A. D. Wentzell. *Random Perturbations of Dynamical Systems*, volume 260 of *Grundlehren der Mathematischen Wissenschaften [Fundamental Principles of Mathematical Sciences]*. Springer-Verlag, New York, 1984.

54. A. Gelman, J. B. Carlin, H. S. Stern, D. B. Dunson, A. Vehtari, and D. B. Rubin. *Bayesian Data Analysis*. CRC Press, 2013.
55. A. Gelman and D. B. Rubin. Inference from iterative simulation using multiple sequences. *Statistical science*, pages 457–472, 1992.
56. S. Ghosal, J. K. Ghosh, and R. V. Ramamoorthi. Consistency issues in Bayesian nonparametrics. In *Asymptotics, Nonparametrics and Time Series: A Tribute*, pages 639–667. Marcel Dekker, 1998.
57. A. Gibbs and F. Su. On choosing and bounding probability metrics. *International Statistical Review*, 70:419–435, 2002.
58. C. González-Tokman and B. R. Hunt. Ensemble data assimilation for hyperbolic systems. *Physica D*, 243:128–142, 2013.
59. G. A. Gottwald and A. Majda. A mechanism for catastrophic filter divergence in data assimilation for sparse observation networks. *Nonlinear Processes in Geophysics*, 20(5):705–712, 2013.
60. G. Grimmett and D. Stirzaker. *Probability and Random Processes*. Oxford University Press, New York, 2001.
61. J. Guckenheimer and P. Holmes. *Nonlinear Oscillations, Dynamical Systems, and Bifurcations of Vector Fields*, volume 42 of *Applied Mathematical Sciences*. Springer-Verlag, New York, 1983.
62. M. Hairer, A. M. Stuart, and J. Voss. Analysis of SPDEs arising in path sampling. Part II: The nonlinear case. *Ann. Appl. Prob.*, 17:1657–1706, 2007.
63. M. Hairer, A. M. Stuart, and J. Voss. Signal processing problems on function space: Bayesian formulation, stochastic PDEs and effective MCMC methods. In *Oxford Handbook of Nonlinear Filtering*, 2010.
64. M. Hairer, A. M. Stuart, J. Voss, and P. Wiberg. Analysis of SPDEs arising in path sampling. Part I: The Gaussian case. *Comm. Math. Sci.*, 3:587–603, 2005.
65. A. Harvey. *Forecasting, Structural Time Series Models and the Kalman Filter*. Cambridge Univ Pr, 1991.
66. W. K. Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1):97–109, 1970.
67. K. Hayden, E. Olson, and E. Titi. Discrete data assimilation in the Lorenz and Dd Navier–Stokes equations. *Physica D: Nonlinear Phenomena*, 2011.
68. D. Higham. An algorithmic introduction to numerical simulation of stochastic differential equations. *SIAM Rev.*, 43(3):525–546 (electronic), 2001.
69. V. H. Hoang, K. J. Law, and A. M. Stuart. Determining white noise forcing from Eulerian observations in the Navier–Stokes equation. *Stochastic Partial Differential Equations: Analysis and Computations*, 2(2):233–261, 2014.
70. I. Hoteit, X. Luo, and D.-T. Pham. Particle Kalman filtering: A nonlinear Bayesian framework for ensemble Kalman filters. *Mon. Wea. Rev.*, 140(2), 2012.
71. I. Hoteit, D.-T. Pham, G. Triantafyllou, and G. Korres. A new approximate solution of the optimal nonlinear filter for data assimilation in meteorology and oceanography. *Mon. Wea. Rev.*, 136(1), 2008.
72. P. Houtekamer and H. Mitchell. A sequential ensemble Kalman filter for atmospheric data assimilation. *Mon. Wea. Rev.*, 129(1):123–137, 2001.
73. A. R. Humphries and A. M. Stuart. Deterministic and random dynamical systems: theory and numerics. In *Modern methods in scientific computing and applications (Montréal, QC, 2001)*, volume 75 of *NATO Sci. Ser. II Math. Phys. Chem.*, pages 211–254. Kluwer Acad. Publ., Dordrecht, 2002.
74. K. Ide, M. Courier, M. Ghil, and A. Lorenc. Unified notation for assimilation: Operational, sequential and variational. *J. Met. Soc. Japan*, 75:181–189, 1997.
75. K. Ide and C. Jones. Special issue on the mathematics of data assimilation. *Physica D*, 230:vii–viii, 2007.
76. M. A. Iglesias, K. J. H. Law, and A. M. Stuart. Evaluation of Gaussian approximations for data assimilation in reservoir models. *Computational Geosciences*, 17:851–885, 2013.
77. A. Jazwinski. *Stochastic Processes and Filtering Theory*, volume 63. Academic Pr, 1970.
78. J. Kaipio and E. Somersalo. *Statistical and Computational Inverse Problems*, volume 160 of *Applied Mathematical Sciences*. Springer-Verlag, New York, 2005.
79. R. Kalman. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 82(1):35–45, 1960.
80. R. Kalman and R. Bucy. New results in linear filtering and prediction theory. *Journal of Basic Engineering*, 83(3):95–108, 1961.
81. E. Kalnay. *Atmospheric Modeling, Data Assimilation and Predictability*. Cambridge, 2003.

82. D. T. B. Kelly, K. J. H. Law, and A. M. Stuart. Well-posedness and accuracy of the ensemble Kalman filter in discrete and continuous time. *Nonlinearity*, 27(10):2579, 2014.
83. M. Kessler, A. Lindner, and M. Sorensen. *Statistical Methods for Stochastic Differential Equations*, volume 124. CRC Press, 2012.
84. P. Kloeden and E. Platen. *Numerical Solution of Stochastic Differential Equations*, volume 23 of *Applications of Mathematics (New York)*. Springer-Verlag, Berlin, 1992.
85. Y. A. Kutoyants. *Statistical Inference for Ergodic Diffusion Processes*. Springer Series in Statistics. Springer-Verlag London Ltd., London, 2004.
86. P. Lancaster and L. Rodman. *Algebraic Riccati Equations*. Oxford University Press, 1995.
87. K. J. H. Law, D. Sanz-Alonso, A. Shukla, and A. M. Stuart. Filter accuracy for chaotic dynamical systems: fixed versus adaptive observation operators. <http://arxiv.org/abs/1411.3113>.
88. K. J. H. Law, A. Shukla, and A. M. Stuart. Analysis of the 3DVAR filter for the partially observed Lorenz '63 model. *Discrete and Continuous Dynamical Systems A*, 34:1061–1078, 2014.
89. K. J. H. Law and A. M. Stuart. Evaluating data assimilation algorithms. *Mon. Wea. Rev.*, 140:3757–3782, 2012.
90. F. Le Gland, V. Monbet, and V.-D. Tran. Large sample asymptotics for the ensemble Kalman filter. 2009.
91. J. Li and D. Xiu. On numerical properties of the ensemble Kalman filter for data assimilation. *Computer Methods in Applied Mechanics and Engineering*, 197(43):3574–3583, 2008.
92. J. S. Liu. *Monte Carlo Strategies in Scientific Computing*. Springer Series in Statistics. Springer, 2001.
93. A. C. Lorenc. Analysis methods for numerical weather prediction. *Quart. J. R. Met. Soc.*, 112(474):1177–1194, 2000.
94. A. C. Lorenc, S. P. Ballard, R. S. Bell, N. B. Ingleby, P. L. F. Andrews, D. M. Barker, J. R. Bray, A. M. Clayton, T. Dalby, D. Li, T. J. Payne, and F. W. Saunders. The Met. Office global three-dimensional variational data assimilation scheme. *Quart. J. R. Met. Soc.*, 126(570):2991–3012, 2000.
95. E. Lorenz. Deterministic nonperiodic flow. *Atmos J Sci*, 20:130–141, 1963.
96. E. Lorenz. The problem of deducing the climate from the governing equations. *Tellus*, 16(1):1–11, 1964.
97. E. Lorenz. Predictability: A problem partly solved. In *Proc. Seminar on Predictability*, volume 1, pages 1–18, 1996.
98. D. G. Luenberger. *Optimization by Vector Space Methods*. John Wiley & Sons, 1968.
99. A. Majda, D. Giannakis, and I. Horenko. Information theory, model error and predictive skill of stochastic models for complex nonlinear systems. *Physica D*, 241:1735–1752, 2012.
100. A. Majda and J. Harlim. *Filtering Complex Turbulent Systems*. Cambridge University Press, 2012.
101. A. Majda, J. Harlim, and B. Gershgorin. Mathematical strategies for filtering turbulent dynamical systems. *Disc. Cont. Dyn. Sys.*, 2010.
102. A. Majda and X. Wang. *Nonlinear Dynamics and Statistical Theories for Geophysical Flows*. Cambridge, Cambridge, 2006.
103. J. Mandel, L. Cobb, and J. D. Beezley. On the convergence of the ensemble Kalman filter. *Applications of Mathematics*, 56(6):533–541, 2011.
104. X. Mao. *Stochastic Differential Equations and Their Applications*. Horwood Publishing Series in Mathematics & Applications. Horwood Publishing Limited, 2008.
105. J. C. Mattingly and A. M. Stuart. Geometric ergodicity of some hypo-elliptic diffusions for particle motions. *Markov Processes and Related Fields*, 8(2):199–214, 2002.
106. J. C. Mattingly, A. M. Stuart, and D. J. Higham. Ergodicity for SDEs and approximations: locally Lipschitz vector fields and degenerate noise. *Stochastic Process. Appl.*, 101(2):185–232, 2002.
107. N. Metropolis, R. Rosenbluth, M. Teller, and E. Teller. Equations of state calculations by fast computing machines. *J. Chem. Phys.*, 21:1087–1092, 1953.
108. S. P. Meyn and R. L. Tweedie. *Markov Chains and Stochastic Stability*. Communications and Control Engineering Series. Springer-Verlag London Ltd., London, 1993.
109. A. Moodey, A. Lawless, R. Potthast, and P. van Leeuwen. Nonlinear error dynamics for cycled data assimilation methods. *Inverse Problems*, 29(2):025002, 2013.
110. L. Nerger, T. Janjic, J. Schröter, and W. Hiller. A unification of ensemble square root Kalman filters. *Mon. Wea. Rev.*, 140(7):2335–2345, 2012.
111. N. Nichols. Data assimilation: aims and basic concepts. In *Data Assimilation for the Earth System*, Editors R. Swinbank, V. Shutyaev, W.A. Lahoz, pages 9–20. Kluwer, 2003.
112. J. Nocedal and S. Wright. *Numerical Optimization*. Springer Verlag, 1999.

113. J. Norris. *Markov Chains*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, Cambridge, 1998.
114. B. Oksendal. *Stochastic Differential Equations*. Universitext. Springer, sixth edition, 2003. An introduction with applications.
115. D. Oliver, A. Reynolds, and N. Liu. *Inverse Theory for Petroleum Reservoir Characterization and History Matching*. Cambridge Univ Pr, 2008.
116. E. Olson and E. Titi. Determining modes for continuous data assimilation in 2D turbulence. *Journal of statistical physics*, 113(5–6):799–840, 2003.
117. E. Ott, B. Hunt, I. Szunyogh, A. Zimin, E. Kostelich, M. Corazza, E. Kalnay, D. Patil, and J. Yorke. A local ensemble Kalman filter for atmospheric data assimilation. *Tellus A*, 56:273–277, 2004.
118. A. Papanicolaou. Stochastic analysis seminar on filtering theory. *arXiv:1406.1936*, 2014.
119. D. F. Parrish and J. C. Derber. The national meteorological centers spectral statistical-interpolation analysis system. *Mon. Wea. Rev.*, 120(8):1747–1763, 1992.
120. G. Pavliotis. *Stochastic Processes*. Springer, 2014.
121. D. Pham, J. Verron, and L. Gourdeau. Singular evolutive Kalman filters for data assimilation in oceanography. *Oceanographic Literature Review*, 45(8), 1998.
122. P. Rebeschini and R. van Handel. Can local particle filters beat the curse of dimensionality? *Ann. Appl. Probab.*, 25, 2809–2866 (2015).
123. S. Reich. A Gaussian-mixture ensemble transform filter. *Q.J.Roy.Met.Soc.*, 138(662):222–233, 2012.
124. C. Robert and G. Casella. *Monte Carlo Statistical Methods*. Springer Texts in Statistics. Springer-Verlag, 1999.
125. B. Rozovskii. *Stochastic Evolution Systems*, volume 35 of *Mathematics and Its Applications (Soviet Series)*. Kluwer Academic Publishers Group, Dordrecht, 1990.
126. D. Sanz-Alonso and A. M. Stuart. Long-time asymptotics of the filtering distribution for partially observed chaotic dynamical systems. *arXiv:1406.1936*, 2014 (to Appear in SIAM J. Uncertainty Quantification)
127. E. Schröder. Ueber iterirte Functionen. *Mathematische Annalen*, 3(2):296–322, 1870.
128. L. Slivinski, E. Spiller, A. Apte, and B. Sandstede. A hybrid particle-ensemble Kalman filter for Lagrangian data assimilation. *Mon. Wea. Rev.*, 145:195–211, 2015.
129. T. Snyder, T. Bengtsson, P. Bickel, and J. Anderson. Obstacles to high-dimensional particle filtering. *Mon. Wea. Rev.*, 136:4629–4640, 2008.
130. E. Sontag. *Mathematical Control Theory*. Springer, 1998.
131. P. Spanos and R. Ghanem. *Stochastic Finite Elements: A Spectral Approach*. Dover, 2003.
132. C. Sparrow. *The Lorenz Equations: Bifurcations, Chaos, and Strange Attractors*, volume 41 of *Applied Mathematical Sciences*. Springer-Verlag, New York, 1982.
133. A. M. Stuart. Inverse problems: a Bayesian perspective. *Acta Numer.*, 19:451–559, 2010.
134. A. M. Stuart and A. R. Humphries. *Dynamical Systems and Numerical Analysis*, volume 2 of *Cambridge Monographs on Applied and Computational Mathematics*. Cambridge University Press, Cambridge, 1996.
135. P. Talagrand, O. Courtier. Variational assimilation of meteorological observations with the adjoint vorticity equation. I: theory. *Quart. J. Royal Met. Soc.*, 113:1311–1328, 1987.
136. A. Tarantola. *Inverse Problem Theory*. SIAM, 2005.
137. R. Temam. *Infinite-Dimensional Dynamical Systems in Mechanics and Physics*, volume 68 of *Applied Mathematical Sciences*. Springer-Verlag, New York, second edition, 1997.
138. M. Tippett, J. Anderson, C. Bishop, T. Hamill, and J. Whitaker. Ensemble square root filters. *Mon. Wea. Rev.*, 131(7):1485–1490, 2003.
139. W. Tucker. The Lorenz attractor exists. *C. R. Acad. Sci. Paris Sér. I Math.*, 328(12):1197–1202, 1999.
140. W. Tucker. A rigorous ODE solver and Smale's 14th problem. *Found. Comput. Math.*, 2(1):53–117, 2002.
141. A. W. Van der Vaart. *Asymptotic Statistics*, volume 3. Cambridge University Press, 2000.
142. P. Van Leeuwen. Particle filtering in geophysical systems. *Mon. Wea. Rev.*, 137:4089–4114, 2009.
143. P. van Leeuwen. Nonlinear data assimilation in geosciences: an extremely efficient particle filter. *Q.J.Roy.Met.Soc.*, 136(653):1991–1999, 2010.
144. P. Van Leeuwen and G. Evensen. Data assimilation and inverse methods in terms of a probabilistic formulation. *Mon. Wea. Rev.*, 124:2892–2913, 1996.
145. M. Viana. *Stochastic Dynamics of Deterministic Systems*, volume 21. IMPA Brazil, 1997.

146. J. Whitaker and T. Hamill. Ensemble data assimilation without perturbed observations. *Mon. Wea. Rev.*, 130(7):1913–1924, 2002.
147. S. Wiggins. *Introduction to Applied Nonlinear Dynamical Systems and Chaos*, volume 2 of *Texts in Applied Mathematics*. Springer-Verlag, New York, 2003.
148. D. Williams. *Probability with Martingales*. Cambridge University Press, Cambridge, 1991.
149. J. Zabczyk. *Mathematical Control Theory: An Introduction*. Springer, 2009.
150. O. Zeitouni and A. Dembo. A maximum a posteriori estimator for trajectories of diffusion processes. *Stochastics*, 20:221–246, 1987.
151. D. Zupanski. A general weak constraint applicable to operational 4DVAR data assimilation systems. *Mon. Wea. Rev.*, 125:2274–2292, 1997.

Index

Symbols

- σ -algebra, 1
- Borel, 1
- 3DVAR, xiii, 51, 83, 84, 95, 99, 103, 104, 106–109, 111, 112, 133, 135, 189–191, 195, 196, 198–200, 221
- 4DVAR, xiii, 54, 65, 72, 73, 75, 84, 119
- strong constraint, 67
- weak constraint, xiii, 54, 67, 73, 119, 128

A

- absorbing set, 13, 31, 50
- allocation, 120
- almost surely, 2
- analysis, 38, 79
- approximate Gaussian filter, 198
- atmospheric sciences, xi, 32, 49, 65
- autocorrelation, 58

B

- background
 - covariance, 37
 - mean, 35, 37
 - penalization, 35, 37, 117
- Bayes's formula, xv, 7, 8, 21, 34, 35, 38, 79–81, 89, 90, 94
- Bayesian, xi–xiii, 33, 44, 45, 51, 54, 65, 68, 75, 88, 95, 161, 185
- Bayesian quality assessment, 45, 95, 194
- Borel
 - σ -algebra, 1
 - set, 1, 2, 12
- Brownian dynamics, 171
- Brownian motion, 152–154, 156, 157, 162, 172, 173, 178–180, 202, 208, 215
- burn-in, 122
- burn-in time, 75, 119

C

- calculus of variations, xi, xiv, 65, 68, 185
- chaos, 32, 92
- conditional distribution, 6, 163
- conditioning theorem, 162

consistency checks, 52

covariance, xv, 80

D

- data, xi–xv, 25, 26, 33, 34, 36, 40, 45, 54, 57, 61, 151–153, 161, 165, 187
 - accumulated, xv, 38, 79, 163, 166, 187
 - sequential, xiii
- data assimilation, xi, 1, 9, 14, 15, 26, 43–45, 49, 51, 53, 87, 153, 161
 - window, 34, 163
- detailed balance, 58–60
- deterministic dynamics, 26, 29, 45, 53, 65, 79, 101, 117, 128, 152, 153, 161, 172, 179, 182, 185, 189
- deterministic signal dynamics, 153
- diffusion process, 75, 155
- diffusivity matrix, 154
- Dirac mass, 5, 88
- distance, 15, 45
 - Hellinger, 16, 17, 22
 - total variation, 16, 17
 - total-variation, 23
- distributed, 2
- dynamical system, xii, 9, 13, 32
 - continuous-time, 9, 11
 - deterministic, xii
 - discrete, 12, 13
 - discrete-time, 9
 - ergodic, 20
 - stochastic, xii, 10, 79

E

- empirical measure, 28, 156–159
- equilibrium point, 11, 52, 158
- ergodic, xv, 20–22, 27–30, 50, 57, 58, 60, 69, 70, 91, 111, 155, 173
 - dynamical system, 20
 - theorem, 20
- event, 2
- expected value, 2
- exponential martingale, 167

F

filter

- approximate Gaussian, 50, 83–85, 87, 95, 105, 108, 111, 187, 189, 194
- bootstrap, 87, 90, 94, 95, 194
- particle, xiii, xiv, 39, 51, 53, 198
- synchronization, 101, 111, 133, 199
- filtering, xii, xiii, 25, 38–40, 43, 44, 51, 79, 99, 106, 151, 166, 172, 187, 198
- algorithm, xiii, xiv, 79, 130, 145, 187, 195, 197, 219
- analysis step, 39, 79–81, 85, 133, 188, 190
- distribution, xiii, xiv, 39, 43, 51, 52, 79, 80, 87, 89–92, 94, 112, 130, 139, 141, 143, 166, 172, 188–190
- prediction step, 38, 39, 80, 83, 85, 86, 88, 188, 190, 192
- program, 130, 137
- update, 79
- fixed point, 9, 21, 30, 47, 96, 97
- Fokker–Planck equation, 155, 156, 166
- forecast skill, 45, 51
- Frobenius norm, xv, 153

G

- Galerkin, 112, 172
- Gaussian, xii, xv, 3, 26
- Gaussian filter
 - approximate, 189, 198
- generator, 155, 156, 166
- geophysical sciences, xi–xiii, 49–51
 - subsurface, 51
- Girsanov formula, xiv, 161, 163–165, 172
- global attractor, 13, 31, 32, 50, 92
- Gronwall, 99, 196
- Gronwall lemma, 13
 - continuous-time, 12
 - discrete-time, 9

H

- histogram, 28, 29, 68, 113, 156, 157, 182
- rank, 52, 112, 113

I

- i.i.d., xv, 8, 25, 26, 34, 35, 38, 46, 59, 80, 85, 113, 116, 120, 124, 139, 152, 211
- identical twin experiments, 51
- independence dynamics sampler, 62, 64, 69–72, 76, 91, 122, 124, 126, 178, 182–184, 213, 215
- innovation, 82, 131, 139, 141, 143, 219
- integral equation, 154
- invariant density, 21, 30
- invariant measure, 20, 21, 27, 32, 57, 155–160
- Itô formula, xiv, 155, 162, 167, 168, 172, 173, 180, 196, 202
- Itô integral, 153, 154, 173
- Itô isometry, 157, 203

K

- Kalman filter, xiii, 49, 79–83, 85, 95–99, 102, 106, 111, 113, 131, 141, 198
- EnKF, 84
- ensemble, xiii, xiv, 51, 83–87, 103, 104, 106–110, 112, 141, 143, 187, 191, 192, 198, 202, 222, 225
- ensemble square-root, 86
- ensemble transform, 86, 141, 203
- ESRF, 111
- ETKF, 86, 112, 225
- ExKF, 84
- extended, xiii, xiv, 51, 83, 84, 103–105, 107–109, 111, 141, 187, 190, 191, 198, 201, 222
- Kalman–Bucy, xiv, 187–189, 194–198, 219
 - square root, 111
- Kalman gain, 82, 131, 139, 141, 219
- Kalman smoother, 53, 54, 56, 65, 175, 176
 - Kalman–Bucy, 198
- Kalman–Bucy smoother, 175
- krigging, 107
- Kullback–Leibler divergence, 23
- Kushner–Stratonovich equation, xiv, 166–168, 172, 187, 188, 198

L

- Langevin
 - overdamped, 171, 221, 222, 225
- likelihood, 8, 22, 35, 39, 90, 91, 93, 107
 - log-likelihood, 62
- localization, 109, 110
- Lorenz model, 147, 154, 198
 - '63, 111, 158, 180, 217, 229
 - '96, 112, 160, 229
 - 63, 31
 - 96, 32
- LU factorization, 74, 184, 198
- Lyapunov function, 22

M

- MAP estimator, 65, 68, 72, 73, 75, 179, 180
- marginal distribution, 6, 44, 160
- Markov chain, xii, 19–22, 25, 57–60, 62, 63, 65, 70, 71, 74, 76, 89, 91, 93, 119, 122, 177, 183, 211
- Markov inequality, 3
- Markov kernel, xv, 19, 63, 79, 91, 92, 94
- Markov process, 154, 158
 - reversible, 171
- maximum a posteriori estimators, 65
- MCMC, xiii, 54, 57, 59, 60, 62, 64, 65, 68–70, 74, 119, 177, 178, 180, 182
- mean, xv
- measurable, 19, 22
- measure
 - Gaussian, xv, 162
 - Lebesgue, 2
 - probability, xv

- metric, 15
 Hellinger, 16, 18, 51, 77
 total variation, 16, 51, 52, 77, 91
- Metropolis–Hastings, 54, 57–62, 77, 177
- mixing time, 75
- model error, 44, 45, 51
- model misspecification, 44
- model–data misfit, 35, 37, 67, 117, 177
- Monte Carlo, 53, 87
 approximation, 85
 hybrid, 75
 Markov chain, xiii, 54, 57, 74
 sampling, 53, 87
- N**
- neuroscience, xii
- noise
 additive, xii, 26
- normalization constant, 8, 49, 54, 81, 164
- O**
- observation operator, 26, 54
- observations, 26, 48, 83, 100, 111, 151, 189
- oceanography, xi, xii, 49
- oil recovery, xi, xii, 49
- Onsager–Machlup functional, 75
- optimal interpolation, 106
- optimization, 54, 65, 68, 72, 73, 75, 119, 128, 175, 180, 185, 217
- Ornstein–Uhlenbeck process, 157, 162, 171
- P**
- particle filter, 79, 87, 88, 90–92, 94, 95, 103, 104, 110, 111
- pathspace, xiv, 161, 162, 168, 177, 178, 213
- pCN, 50, 62–64, 69, 71–76, 124, 126, 178, 182–185, 215
- perturbed observations, 85, 86, 104, 108, 111, 139, 141
- polynomial chaos, 172
- posterior, 21
 log-posterior, 35, 37, 38, 61, 62, 65
- posterior consistency, 44, 47, 170, 185
 Bayesian, 44
- posterior distribution, xv, 35, 37, 45, 46, 48, 51, 54, 57, 61, 71, 72, 75, 79, 111, 117, 119, 122, 128, 163, 177, 213
- preallocation, 117, 119, 122
- precision, xv, 6, 55, 80, 81, 179, 195
- prediction, 38, 79
- prior, 21, 34
- prior distribution, xv
- probability
 conditional, xii, xiii, 7
 marginal, 6
 triple, 1
- probability density function, xv, 1, 22, 28, 49, 50, 59, 68, 166
- probability measure, xv, 15, 16, 18, 19, 22, 33, 38
- proposal, 60, 62, 64, 92, 94, 103, 119, 120, 122, 124, 126, 145, 177–179, 183, 185, 211, 213, 215
- covariance, 61, 177
- Gaussian, 60, 63, 72
- independence dynamics sampler, 62
- index, 111
- local, 178
- optimal, 79, 94, 95, 104, 109, 111, 225
- pCN, 62–64, 178
- pCN dynamics sampler, 64
- RWM, 60, 62
- standard, 95, 104, 108, 225
- variance, 61, 64, 126, 177, 185
- pushforward, 2, 36, 38, 39, 92, 172
- R**
- Radon–Nikodym derivative, 88, 161, 163, 164
- random variable, 1
 Gaussian, 3
 normal, 3
- random walk Metropolis, 60, 62, 68, 77, 119, 177, 178, 182, 211
 RWM, 60
- reanalysis, 52
- rough paths, 172
- RWM, 60
- S**
- seed, 116, 118, 128, 207
- semigroup, 11, 22, 165
- sequential importance resampling, 88
- shadowing, 112
- signal, xiii–xv, 25, 26, 33, 34, 40, 45, 51, 54, 57, 61, 151–153, 156, 161, 163, 165, 187
- signal estimation quality assessment, 45, 95, 194
- smoothing, xii, xiii, 25, 38–41, 43, 44, 48, 53, 151, 172
 algorithm, xii
 distribution, xiv, 34, 35, 37, 39, 117, 119, 122, 213
- solution operator, 11, 26, 44
- stochastic differential equations, xiv, 50, 151, 154, 161
 reversible, 156, 171
- stochastic dynamics, xiv, 25, 26, 28, 35–37, 39–41, 43–45, 52–55, 57, 61–66, 68–70, 76, 77, 79, 116, 122, 152, 172, 176–180, 182, 185, 192, 213
- stochastic partial differential equations, xiv, 166
- synchronization, 101, 111, 133, 199
- T**
- truth, 44, 45, 47, 98

V

variance inflation, 100, 101, 108–110
variational method, xi, xiii, xiv, 54, 65, 68, 72, 119,
175, 179, 180, 185

W

weak convergence, 3, 58
weather forecasting, xi–xiii, 44, 45, 49,
51

well posed, 25

well-posed, xii, 172

Woodbury matrix identity, 82, 193

Z

Zakai, 167
Zakai equation, xiv, 166, 172, 187, 188, 198