Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Mathematical Sciences

Mads Adrian Simonsen

Approximate Filtering Approaches for Switching Linear Dynamical Systems

Master's thesis in Applied Physics and Mathematics
Supervisor: Jo Eidsvik
June 2022

NTNU
Norwegian University of
Science and Technology

Mads Adrian Simonsen

Approximate Filtering Approaches for Switching Linear Dynamical Systems

Master's thesis in Applied Physics and Mathematics
Supervisor: Jo Eidsvik
June 2022

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Mathematical Sciences

NTNU
Norwegian University of Science and Technology

Preface

This thesis concludes my Master of Science degree in Industrial mathematics at the Norwegian University of Science and Technology (NTNU). First of all, I would like to thank my supervisor Jo Eidsvik for support and guidance, and in limiting the scope of the work. Secondly, I would like to thank Ariel Skår, for all her love and support throughout the years. I would also thank my friend Rasmus Erlemann for insipring me to specialize in statistics. Finally, I am grateful for all the interesting discussions with my brother Une André Simonsen regarding the topic of this thesis.

Mads Adrian Simonsen
Trondheim, June 2022

Abstract

The state-observation model is a dynamic Bayesian network (DBN) that contains two set of variables that evolves over time; a hidden set of variables, and an observed set of variables. The most common state-observation models are the hidden Markov model (HMM) and the linear dynamical system (LDS) where the well known Kalman filter is used for inference. Switching linear dynamical system (SLDS) models are an extension of LDSs, that contain both discrete and continuous hidden variables, whereas the observation variable is continuous. The main idea behind SLDSs is that we can break down complex dynamical behavior into more comprehensible pieces. For instance, we can categorize the motions of an aircraft into the states climbing, descending and maneuver.

The problem with SLDS is that due to the DBN containing both discrete and continuous variables, when we compute the posterior probabilities conditioned on the observations, a process called *filtering*, we end up with a Gaussian mixture which increases exponentially at each time step. Therefore we must look for alternative approximate filtering approaches.

In this thesis we develop the statistical framework for SLDSs and DBNs generally, where the main focus is filtering. We then compare approximate filtering approaches where the deterministic approaches solves the problem by constraining the size of the mixture while preserving the first two moments, (mean and variance), and the stochastic approach solves the problem by particle filtering. Finally we show an application of the SLDS, where the goal is to detect when we are walking, and when we are on the bus given GPS (Global Positioning System) data.

Sammendrag

State-observation model er et dynamisk Bayesiansk nettverk (DBN) som inneholder to mengder stokastiske variabler som utvikler seg over tid; en skjult mengde og en observert mengde. De mest kjente "state-observation" modellene er *hidden Markov model* (HMM) og lineære dynamiske system (LDS), hvor den velkjente Kalmanfilter-metoden er brukt for statistisk inferens. *Switching linear dynamical system* (SLDS) modeller er en utvidelse av LDSer, som inneholder både diskrete og kontinuerlige skjulte variabler, og hvor den observerte variabelen er kontinuerlig. Hovedidéen bak SLDSer er at vi kan bryte ned kompleks dynamisk atferd i mer forståelige stykker som vi kan kategorisere, f.eks. så kan vi kategorisere bevegelsene til et fly inn følgende tilstander: letter, senker og svinger.

Problemet med SLDS er at siden DBNet inneholder både diskrete og kontinuerlige variabler, når vi regner ut posterioirisannsynlighetene gitt observasjonene, en prosess som kalles *filtrering*, så ender vi opp med en Gaussisk miksfordeling, som øker eksponensielt hvert tidssteg. Vi må derfor søke etter alternative approksimasjoner for filtreringen.

I denne oppgaven, utleder vi det statistiske rammeverket for SLDSer og DBNer generelt, hvor hovedfokuset er filtrering. Deretter sammenligner vi approksimative filtreringsmetoder hvor de deterministiske metodene løser problemet ved å begrense størrelsen til miksen samtidig som vi bevarer de to første momentene, (forventingsverdi og varians), mens den stokastiske metoden løser problemet med en teknikk som kalles partikkelfiltrering ("particle filtering").

Til slutt viser vi en anvendelse av SLDS, hvor målet er å detektere når vi går, og når vi er på bussen, gitt GPS-data (Global Positioning System).

Contents

Chapter 1

Introduction

1.1 Background and motivation

When we want to understand a complex dynamical system, it helps to break it down into smaller chunks. If we wish to learn the chord progression of a song, we may first want to break it down into sections such as verse, pre-chorus and chorus, and study each section individually, but also the relationship between the sections. Or we may be interested in studying the motion of an aircraft. Then it helps to break the motion down into specific groups of maneuver such as climbing, descending and turning as well as constant uniform motion, and the relationships between each maneuver.

Switching Linear Dynamical System (SLDS) models are probabilistic models that combines each "chunk" which we call *mode* or *switching state*. Each mode is modeled as a Linear Dynamical System (LDS) where we have a hidden/latent (continuous) *dynamic state* and an observed (measured) state. In the aircraft motion example, the hidden dynamic state may be the exact position of the aircraft and its velocity, and the observed state may be global positioning system (GPS) data collected every tenth second.

SLDSs are a variant of the more general *state-observation models*, which combines a dynamic hidden model and an observation model which we can represent using the language of graphs. These graphs are called Bayesian networks (BNs). BNs consist of nodes where each node represents a random variable with with a set of possible outcomes. The BN also have edges between some of the nodes, that represent the probabilistic dependencies between the variables. An example is shown in Figure 1.1, where we have three variables *Sunny*, *Ice cream* and *Sunburn* where each variable may be true or false. That is, if *Sunny* is true, it means that it is sunny outside at a given day, if *Ice cream* is true, it means that you were eating ice cream that day, and if *Sun burn* is true, it means that you got a sunburn. In this BN we assume that getting a sunburn depends on whether it is sunny outside or not. Similarly, we may assume that the probability of eating an ice cream that day changes depending on whether it is sunny or not.

The SLDS is dynamical which means it incorporates time, or have a certain "ordering". The graphical structure of SLDS models is therefore a BN that expands as time passes, that we call a dynamic Bayesian network (DBN), shown in Figure 1.2. Here the

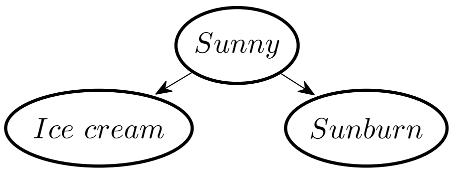


Figure 1.1: BN example. Each node represents a random variable with possible outcomes being *true* or *false*. The arrows are called edges and encodes the dependence between the variables.

hidden states are $S^{(t)}$ and $\mathbf{Y}^{(t)}$ at time step t, where the former is the discrete switching state and the latter is the dynamic (continuous) state, and $\mathbf{O}^{(t)}$ is the observed state.

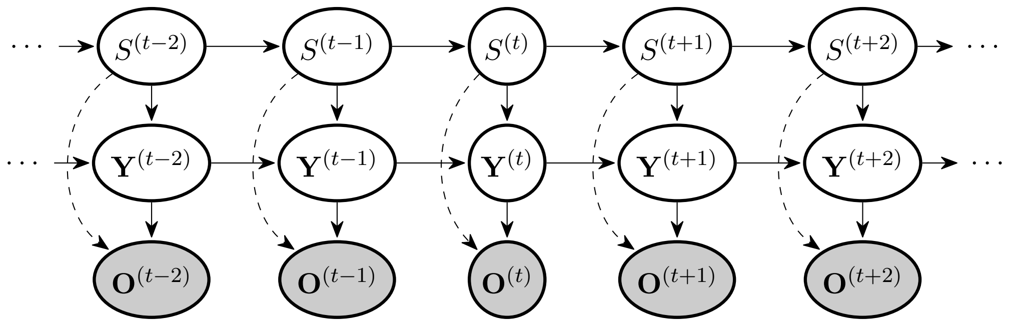


Figure 1.2: SLDS, where $S^{(t)}$ is the switching state, $\mathbf{Y}^{(t)}$ the dynamical state, and $\mathbf{O}^{(t)}$ the observation at time t. Dashed edges are optional dependencies.

We see that the DBN is simply a BN with the same variables in each time step. Other state-observation models include the hidden Markov model (HMM) which has been successfully applied in various areas such as speech recognition (Rabiner, 1989), gesture recognition (Starner & Pentland, 1997) and bioinformatics (Li & Stephens, 2003).

SLDS models have applications in macroeconometrics, where the model has the name *Switching regime models* (Durlauf & Blume, 2016). Here the goal is to detect when we are in a financially stable period with economic growth versus an unstable period with economic decay. SLDSs are also used in *Stop-and-Go* situations where we want to detect whether a car is not moving, driving at constant velocity or driving at constant acceleration (Kaempchen et al., 2004). Other applications include classification of tempo of a musical piece (Gu & Raphael, 2012) or prediction of human motion trajectory (Rudenko et al., 2020).

In this thesis, after exploring some possible approximations to the filtering problem, we show an application where the goal is to detect whether we are on the bus or walking when on a trip, where we use GPS data as observations.

1.2 Outline

In Chapter 2, we start by introducing BNs and their most important properties with regards to conditional independence. Then we introduce DBNs and common variants of state-observation models, and their representations.

In Chapter 3, we tackle the inference problem of state-observation models. We take advantage of the conditional independence properties of BNs to efficiently compute the probability distribution over the hidden states conditioned on the observation data. The procedure is a recursive algorithm called the *forward algorithm*. We show how to apply the forward algorithm to HMMs, LDSs, and finally SLDSs. Then we show alternative approximations that reduces the running time of the algorithm and compare their performances.

In Chapter 4, we derive the equations of the motion dynamics in a 2-dimensional plane and show an application of the SLDS for the bus and walk-example, and in Chapter 5, we summarize and suggest possible future work.

1.3 Abbreviations and notations

There are several technical terms within BNs and SLDSs, and we summarize the terminology and notation used in this thesis. Table 1.1 shows a list abbreviations, and Table 1.2 to 1.7 show the mathematical notations used.

Table 1.1: List of abbreviations

Abbreviation	Meaning
BN	Bayesian network
CLG	Conditional linear Gaussian
CPD	Conditional probability distribution
CPT	Conditional probability table
DAG	Directed acyclic graph
DBN	Dynamic Bayesian network
EKF	Extended Kalman filter
GPS	Global positioning system
HMM	Hidden Markov model
KF	Kalman filter
LDS	Linear dynamical system
PDF	Probability density function
PMF	Probability mass function
SLDS	Switching linear dynamical system

Table 1.2: List of general notations

Symbol	Meaning
$\mathcal{X}$	The set of all random variables in a given BN
$P(S = s)$	PMF of discrete random variable S
$p_{\mathbf{Y}}(\mathbf{y})$	PDF of continuous random vector $\mathbf{Y}$
$f(\mathbf{y}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$	PDF of $\mathbf{Y}$ with multivariate Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$
$\mathcal{P}(\mathbf{X} = \mathbf{x})$	PMF, PDF or a combination depending on $\mathbf{X}$, conveniently used when $\mathbf{X} = \{S, \mathbf{Y}\}$, where S is discrete and $\mathbf{Y}$ is continuous
$\mathcal{P}(\mathbf{x})$	Shorthand notation for $\mathcal{P}(\mathbf{X} = \mathbf{x})$
$\mathcal{P}(\mathbf{X})$	Shorthand notation for $\mathcal{P}(\mathbf{X} = \mathbf{x})$ for all possible $\mathbf{x}$
$\mathrm{Pa}(\mathbf{X})$	The parents of $\mathbf{X}$, i.e. the set of random variables that $\mathbf{X}$ directly depends on
$\mathrm{NonDesc}(\mathbf{X})$	The non-descendants of $\mathbf{X}$, i.e. the subset of $\mathcal{X}$ that are not descendants of $\mathbf{X}$
$(\mathbf{X} \perp \mathbf{Y} \mid \mathbf{Z})$	$\mathbf{X}$ and $\mathbf{Y}$ are conditionally independent given $\mathbf{Z}$

Table 1.3: List of notations in state-observation models

Symbol	Meaning
t	Natural number denoting the time step of DBNs
$\mathcal{X}^{(t)}$	The set of all random variables in a given DBN at time t
$\mathbf{X}^{(t)}$	(Hidden/latent) state variable(s) at time t,
$\mathbf{O}^{(t)}$	Observed variable(s) at time t
$\mathbf{o}^{(1:t)}$	Observation sequence $\{\mathbf{O}^{(1)} = \mathbf{o}^{(1)}, \dots, \mathbf{O}^{(t)} = \mathbf{o}^{(t)}\}$
$\mathcal{P}(\mathbf{X}' \mid \mathbf{X})$	Transition model $\mathcal{P}(\mathbf{X}^{(t+1)} \mid \mathbf{X}^{(t)})$ for $t = 1, 2, \dots$
$\mathcal{P}(\mathbf{O} \mid \mathbf{X})$	Observation model $\mathcal{P}(\mathbf{O}^{(t)} \mid \mathbf{X}^{(t)})$ for $t = 1, 2, \dots$
$\sigma^{(t)}(\mathbf{x})$	Filtered estimate $\mathcal{P}(\mathbf{X}^{(t)} = \mathbf{x} \mid \mathbf{o}^{(1:t)})$
$\sigma^{(\cdot t+1)}(\mathbf{x})$	Predicted estimate $\mathcal{P}(\mathbf{X}^{(t+1)} = \mathbf{x} \mid \mathbf{o}^{(1:t)})$

Table 1.4: List of notations in HMMs

Symbol	Meaning
$S^{(t)}$	(Switching) state variable with possible outcomes $1, \ldots, M$
$O^{(t)}$	Observation variable with possible outcomes $1, \ldots, L$
$\boldsymbol{\pi}$	Initial probability vector $P(S^{(1)}) = (\pi_i) \in \mathbb{R}^M$ where $\pi_i = P(S^{(1)} = i)$
$\mathbf{P}$	Transition probability matrix $P(S' \mid S) = (p_{ij}) \in \mathbb{R}^{M \times M}$ where $p_{ij} = P(S' = j \mid S = i)$
$\mathbf{E}$	Emission probability matrix $P(O \mid S) = (e_{ij}) \in \mathbb{R}^{M \times L}$ where $e_{ij} = P(O = j \mid S = i)$ for $i = 1, \ldots, M$ and $j = 1, \ldots, L$

Table 1.5: List of notations in LDSs

Symbol	Meaning
$\mathbf{Y}^{(t)}$	(Dynamic) state variable with support $\mathbb{R}^N$
$\mathbf{O}^{(t)}$	Observation variable with support $\mathbb{R}^L$
$\mathcal{N}(\boldsymbol{\nu}, \boldsymbol{\Gamma})$	Probability distribution of initial distribution model $p_{\mathbf{Y}^{(1)}}(\mathbf{y}^{(1)})$
$\mathcal{N}(\mathbf{A}\mathbf{y} + \mathbf{b}, \mathbf{Q})$	CPD of the transition model $p_{\mathbf{Y}'}(\mathbf{y}' \mid \mathbf{y})$
$\mathcal{N}(\mathbf{C}\mathbf{y} + \mathbf{d}, \mathbf{R})$	CPD of the observation model $p_{\mathbf{O}}(\mathbf{o} \mid \mathbf{y})$

Table 1.6: List of notations in SLDSs

Symbol	Meaning
$S^{(t)}$	Switching state variable with possible outcomes $1, \ldots, M$
$\mathbf{Y}^{(t)}$	Dynamic state variable with support $\mathbb{R}^N$
$\mathbf{O}^{(t)}$	Observation variable with support $\mathbb{R}^L$
$\boldsymbol{\pi}$	Initial probability vector $P(S^{(1)}) = (\pi_i) \in \mathbb{R}^M$ where $\pi_i = P(S^{(1)} = i)$
$\mathbf{P}$	Transition probability matrix $P(S' \mid S) = (p_{ij}) \in \mathbb{R}^{M \times M}$ where $p_{ij} = P(S' = j \mid S = i)$
$\mathcal{N}(\boldsymbol{\nu}_{s^{(1)}}, \boldsymbol{\Gamma}_{s^{(1)}})$	Initial distribution model $p(\mathbf{y}^{(1)} \mid s^{(1)})$
$\mathcal{N}(\mathbf{A}_{s'}\mathbf{y} + \mathbf{b}_{s'}, \mathbf{Q}_{s'})$	CPD of the transition model $p_{\mathbf{Y}'}(\mathbf{y}' \mid s', \mathbf{y})$
$\mathcal{N}(\mathbf{C}_s\mathbf{y} + \mathbf{d}_s, \mathbf{R}_s)$	CPD of the observation model $p_{\mathbf{O}}(\mathbf{o} \mid s, \mathbf{y})$

Table 1.7: List of notations in EKFs

Symbol	Meaning
$\boldsymbol{\epsilon}^{(t)}$	Alternative representation of the random noise added to the transition model such that $\boldsymbol{\epsilon}^{(t)} \sim \mathcal{N}(\mathbf{0}, \mathbf{Q})$
$\mathbf{f}(\mathbf{y}^{(t)}, \boldsymbol{\epsilon}^{(t+1)})$	Nonlinear function for the dynamic state $\mathbf{Y}^{(t+1)}$
$\nabla \mathbf{f}^{\mathsf{T}}(\mathbf{y}^{(t)}, \boldsymbol{\epsilon}^{(t+1)})$	The Jacobian of the nonlinear function $\mathbf{f}$

Chapter 2

Model

In Section 2.1, we define BNs, how we graphically represent them and their distributional properties. In Section 2.2, we define DBNs, the temporal variant of BNs with a set of additional assumptions. Furthermore, we introduce a special representation of DBNs which is called state-observation models. We include some common examples of state-observation models such as HMMs, LDSs and SLDSs.

2.1 Bayesian Networks

A BN $\mathcal{B} = (\mathcal{X}, \mathcal{E})$ is a directed acyclic graph (DAG) where the nodes $\mathcal{X} = \{X_1, \dots, X_n\}$ represent random variables, and the directed edges $\mathcal{E}$ going from one node to another represent probabilistic dependence between the variables, i.e. if $(X_i \to X_j) \in \mathcal{E}$, then the variable X_j depends on X_i. Conversely, X_i depends on X_j, but when building the model, we typically choose the direction of an edge to represent a causation, e.g. sunny weather ($\mathit{Sunny} = 1$) may cause sunburn ($\mathit{Sunburn} = 1$), which gives us the edge $\mathit{Sunny} \to \mathit{Sunburn}$. Being acyclic means that when following the direction of the edges from node to node, we can never form a closed loop, e.g. the graph $X_1 \to X_2 \to X_3 \to X_1$ is not a DAG. Pedigree charts or family trees are examples of DAGs, where each node represent a family member, and the edge $(X_i \to X_j)$ means that X_i is a parent of X_j, and that X_j is a child of X_i. Intuitively, the family tree must be acyclic as no person can be its own ancestor.

Because of the close relationship between family trees and DAGs, we stick with the family tree terminology. That is, the parents of $X_i \in \mathcal{X}$ are all the variables that have an edge going directly to X_i, which we denote by $\mathrm{Pa}(X_i)$. If $X_1, \dots, X_N$ are topologically sorted (parents come before children), and there exists a directed path from X_i to X_k, e.g. $X_i \to X_{i+1} \to \cdots \to X_k$, we say that X_i is an ancestor of X_k, and X_k is a descendant of X_i. We also define the non-descendants of X_i to be all variables in $\mathcal{X}$ that are not descendants of X_i, and denote it by $\mathrm{NonDesc}(X_i)$. The probability distribution over the network is defined by the conditional probability distributions (CPDs) for each variable

X_i conditioned on its parents $\mathrm{Pa}(X_i)$:

$$\mathcal{P}(X_i = x_i \mid \mathrm{Pa}(X_i) = \mathrm{pa}(X_i)), \quad i = 1, \ldots, n, \tag{2.1}$$

where $\mathrm{pa}(X_i)$ denotes some outcome of the parents of X_i.

Consider the BN $\mathcal{B}^{\text{sprinkler}}$ that consist of the four random variables *Cloudy* (C), *Sprinkler* (S), *Rainy* (R) and *Wet grass* (W), shown in Figure 2.1, where the sample space of each variable is $\{0, 1\}$ where 0 means false and 1 means true. That is, $C = 0$ means no clouds (or sunny), and $C = 1$ means that it is cloudy. The outcome $S = 1$ means that the sprinkler is running, $W = 1$ means that the grass is wet and $R = 1$ means that it is rainy outside. Here we have for instance assumed that cloudy weather may cause rain indicated by the edge $C \to R$, and the sprinkler and rainy weather may cause wet grass, indicated by $S \to W$ and $R \to W$, respectively. If we think of cloudy weather more as a season, it makes sense that the sprinkler is more likely to run during hot seasons ($C = 0$), giving us the edge $C \to S$.

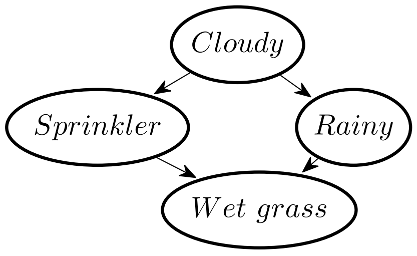


Figure 2.1: Simple example of the BN, $\mathcal{B}^{\text{sprinkler}}$, adapted from Russell and Norvig (2021). The nodes represent random variables, and the edges represent the probabilistic dependencies between the variables.

Since the variables in $\mathcal{B}^{\text{sprinkler}}$ are discrete, we can write the CPDs given by Expression (2.1) as *conditional probability tables* (CPTs) as shown in Table 2.1. In Table 2.1a we see that it is more probable having a cloudy weather than sunny, i.e. $P(C = 1) = 0.6$, and in Table 2.1d in the second row we see that given that the sprinkler is not running but it is rainy, it is more probable of the grass being wet than dry, i.e. $P(W = 1 \mid S = 0, R = 1) = 0.9$. We also notice that each row sum to one, as it is guaranteed for each variable to be either true or false (1 or 0).

Table 2.1: CPTs for the BN $\mathcal{B}^{\text{sprinkler}}$. In each table, the probability of the child node is shown for each possible outcome for each possible outcome of its parents. As C has no parents, (a) only shows the marginal probabilities.

(a) $P(C)$

C = 0	C = 1
0.4	0.6

(b) $P(S \mid C)$

C	S = 0	S = 1
0	0.4	0.6
1	0.9	0.1

(c) $P(R \mid C)$

C	R = 0	R = 1
0	0.9	0.1
1	0.3	0.7

(d) $P(W \mid S, R)$

S	R	W = 0	W = 1
0	0	1	0
0	1	0.1	0.9
1	0	0.1	0.9
1	1	0.01	0.99

All BNs encode information about conditional independence between the variables. That is, each child (variable) is conditional independent of any of its non-descendants given the outcome of its parents. Here we can think of the family tree again. Alice wants to figure out the probability of her grandfather having a certain pair of genes for eye color, and already knows what pair of genes her parents have. Then any knowledge about Alice's own pair of genes for her eye color will not give any additional information about her grandfather's. That is, Alice's and her grandfather's pair of genes for eye color are conditionally independent given information about her parents' pair of genes. In general, a BN with variables $X_1, \ldots, X_n$ encodes the property:

$$(X_i \perp \text{NonDesc}(X_i) \mid \text{Pa}(X_i)), \quad i = 1, \ldots, n, \tag{2.2}$$

which says that X_i is conditionally independent of its non-descendants given its parents. Here the symbol $\perp$ makes an easy shorthand notation for denoting independence. It is equivalent to writing

$$\mathcal{P}(X_i = x_i \mid \text{NonDesc}(X_i), \text{Pa}(X_i)) = \mathcal{P}(X_i = x_i \mid \text{Pa}(X_i)), \quad i = 1, \ldots, n, \tag{2.3}$$

for all x_i in the sample space of X_i and for all possible outcomes of $\text{NonDesc}(X_i)$ and $\text{Pa}(X_i)$. Conditional independence is symmetric, which means that for the random variables variables X, Y and Z, we have that $(X \perp Y \mid Z)$ is equivalent to $(Y \perp X \mid Z)$. This is intuitive and may be shown as follows:

$$\begin{aligned}
\mathcal{P}(Y = y \mid X = x, Z = z) &= \frac{\mathcal{P}(X = x, Y = y, Z = z)}{\mathcal{P}(X = x, Z = z)} \\
&= \frac{\mathcal{P}(Z = z)\mathcal{P}(Y = y \mid Z = z)\mathcal{P}(X = x \mid Y = y, Z = z)}{\mathcal{P}(Z = z)\mathcal{P}(X = x \mid Z = z)} \\
&= \frac{\mathcal{P}(Y = y \mid Z = z)\mathcal{P}(X = x \mid Z = z)}{\mathcal{P}(X = x \mid Z = z)} \\
&= \mathcal{P}(Y = y \mid Z = z),
\end{aligned}$$

for valid outcomes x, y and z, where we in the third equality used that $(X \perp Y \mid Z)$.

Consider the BN $\mathcal{B}^{\text{sprinkler}}$, then Expression (2.2) tells us that for each variable we have:

$$\begin{aligned}
(C \perp \emptyset \mid \emptyset) &\quad \text{No information, } C \text{ has no non-descendants or parents,} \\
(S \perp R \mid C) &\quad S \text{ and } R \text{ are conditionally independent given } C, \\
(R \perp S \mid C) &\quad \text{Equivalent to above due to symmetry,} \\
(W \perp C \mid S, R) &\quad W \text{ and } C \text{ are conditionally independent given } S \text{ and } R,
\end{aligned} \tag{2.4}$$

where $\emptyset$ is the empty set. In other words, the last line of Expression (2.4) claims that knowing whether or not the sprinkler is running and if it is raining, any information about the grass being wet gives no additional information to the probability of the sky being cloudy. Conversely, knowing whether or not the sprinkler is running and if it is raining, any information about the sky being cloudy gives no extra additional information to the probability of the grass being wet.

Assume a BN with topologically sorted variables $\mathcal{X} = \{X_1, \ldots, X_n\}$. Using the chain rule, the joint probability of all the variables in the BN is equal to

$$P(\mathcal{X}) = P(X_1) \prod_{i=2}^{n} P(X_i \mid X_1, \ldots, X_{i-1}) = \prod_{i=1}^{n} P(X_i \mid \text{Pa}(X_i)), \tag{2.5}$$

where we in the last equality used the conditional independence property given by Expression (2.2), since $X_1, \ldots X_{i-1} \in \text{NonDesc}(X_i) \cup \text{Pa}(X_i)$ and $\text{Pa}(X_i) \subseteq \{X_1, \ldots, X_{i-1}\}$. This factorization of the joint probability distribution is called the *Bayesian chain rule.*

The most common task we wish to solve using BNs is inference. In $\mathcal{B}^{\text{sprinkler}}$, we know the probability of the grass being wet given conditions of the sprinkler and rainy weather, but we do not know the probability of the sprinkler is running given that the grass is wet. This can easily be computed using Bayes' rule and the law of total probability

$$\begin{aligned}
&P(S = 1 \mid W = 1) = \frac{P(S = 1, W = 1)}{P(W = 1)} \propto P(S = 1, W = 1) \\
&= \sum_{c=0}^{1} \sum_{r=0}^{1} P(C = c, S = 1, R = r, W = 1) \\
&= \sum_{c=0}^{1} \sum_{r=0}^{1} P(C = c, S = 1, R = r, W = 1) \\
&= \sum_{c=0}^{1} P(C = c) P(S = 1 \mid C = c) \sum_{r=0}^{1} P(R = r \mid C = c) P(W = 1 \mid S = 1, R = r) \\
&= 0.4 \cdot 0.6(0.9 \cdot 0.9 + 0.1 \cdot 0.99) + 0.6 \cdot 0.1(0.3 \cdot 0.9 + 0.7 \cdot 0.99) \\
&= 0.27594,
\end{aligned}$$

which we must divide by the normalizing constant

$$P(W = 1) = \sum_{s=0}^{1} P(S = s, W = 1) = P(S = 0, W = 1) + 0.27594 = 0.63054,$$

that is,

$$P(S = 1 \mid W = 1) = 0.44.$$

When computing the normalizing constant, we also found the marginal probability of the grass being wet.

We mentioned the simplest conditional independence property of BNs given by Expression (2.2), which is used for the Bayesian chain rule in Expression (2.5). However, there may be more conditional independencies encoded in the BN. The following addresses how we can determine these conditional independencies by analyzing the graphical structure of the BN. Most of the derivation and terminology are taken from Koller and Friedman (2009, pp. 68–74) which provides excellent literature on the topic. Other sources include Pearl (1988, pp. 116–122) and Russell and Norvig (2021, pp. 436–438).

First, we define a *trail* to be any sequence of nodes in the graph where all adjacent nodes in the sequence have an edge between them. For instance, in $\mathcal{B}^{\text{sprinkler}}$, there exists two trails between C and R; $C \to R$ and $C \to S \to W \leftarrow R$. To avoid dealing with unnecessary special cases, we assume in this thesis that all BNs are connected DAGs, meaning there always exists a trail between any two nodes in the graph.

If we consider a graph consisting of only two nodes, they must be dependent, by assumption of the edge between them, i.e. that one variable cause a potential change in the other. Now, consider the variables $\mathcal{X} = \{X_1, X_2, X_3\}$. Then there are four possible trails between the variables, which is shown in Figure 2.2. The first trail, $X_1 \to X_2 \to X_3$

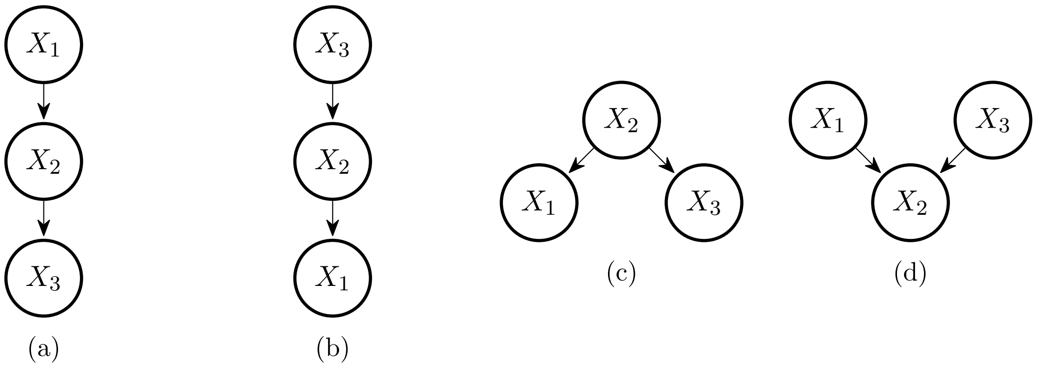


Figure 2.2: Four possible two-edge trails in BNs from X_1 to X_3, where (a) represents a causal trail, (b) an evidential trail, (c) a common cause, and (d) a common effect.

is the one we are familiar with, X_1 causes X_2 which again causes X_3. Consider the BN $\mathcal{B}^{\text{sprinkler}}$, but now without the sprinkler, leaving us with the graph $C \to R \to W$. Assume that we observe that it rains ($R = 1$), then any information about whether or not it is cloudy, does not give any additional information about the grass being wet. Knowing that it is rainy is enough to determine the probability that the grass is wet. On the contrary, if we do not observe R, then any information about clouds would be very helpful, as we may use that information to influence our belief about whether or not it is rainy, and use this belief to further influence our belief of whether or not the grass is wet.

The second trail, is when we flip things around and go backwards against the causal flow. Using the same example (without the sprinkler), we have the trail $W \leftarrow R \leftarrow C$. Assume that we observe R. Then any information about the grass being wet is useless when we are interested in whether or not it is cloudy, as we already have a superior evidence from R. If we do not observe R, then any information about the grass being wet is useful, because it gives us some evidence about whether it or not it is rainy, which again gives us evidence of whether or not it is cloudy.

The third trail, where we have a common cause, we can consider $\mathcal{B}^{\text{sprinkler}}$, where we have the trail $S \leftarrow C \rightarrow R$. Assume that we observe C, then knowing whether or not if the sprinkler is running, does not affect our belief about whether or not it is rainy, since R depends on C, and C is what directly causes R, whereas S is only evidence of what C may be, which is useless. However, when we do not observe C, then the evidence of S is useful, because then we can update our belief of whether or not it is cloudy, and this influences our belief about whether or not it is rainy.

The fourth and final trail, where we have a common effect, we can consider $\mathcal{B}^{\text{sprinkler}}$ but without C. Here we may assume that the sprinkler is ran at random independent of the season. Then we have the trail $S \rightarrow W \leftarrow R$. Observing W, gives us some evidence of what S and R may be. If $W = 1$, then we know that the grass is wet, then we have evidence that it either rained, or the sprinkler ran (or both). Learning that the sprinkler in fact was not running, causes us to conclude that it must be rainy, as we had evidence that at least one of R and S must have caused wet grass. On the contrary, if we do not observe W, then we have no evidence to help us telling whether the grass is wet or not. Even if we know S, then it does provide any extra information whether or not it is rainy, as it runs at random, and we have no evidence of their common effect, W.

To summarize, when we have an indirect influence from X_1 to X_3 through X_2, we say that the trail $X_1 \rightleftharpoons X_2 \rightleftharpoons X_3$ is *active*. Here $\rightleftharpoons$ indicates an edge either pointing left or right. For two-edged trails, we have justified the following result:

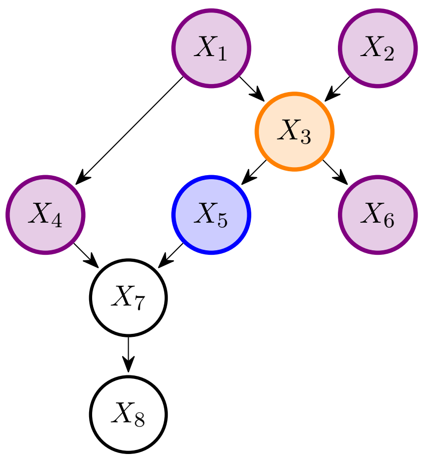
- Causal trail $X_1 \rightarrow X_2 \rightarrow X_3$: active if and only if X_2 is not observed.
- Evidential trail $X_1 \leftarrow X_2 \leftarrow X_3$: active if and only if X_2 is not observed.
- Common cause $X_1 \leftarrow X_2 \rightarrow X_3$: active if and only if X_2 is not observed.
- Common effect $X_1 \rightarrow X_2 \leftarrow X_3$: active if and only if X_2 is observed.

This can be generalized to a case with a longer trail $X_1 \rightleftharpoons \cdots \rightleftharpoons X_n$. Let $\mathbf{Z}$ be a subset of the variables in the graph that are observed. Then the trail is active if the following two conditions hold:

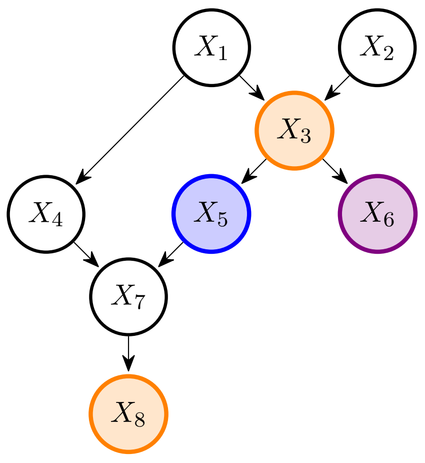
- Whenever we have the structure $X_{i-1} \rightarrow X_i \leftarrow X_{i+1}$ (called a *v-structure*), then X_i or any of its descendants exists in $\mathbf{Z}$.
- No other variables along the path exists in $\mathbf{Z}$.

Here, the only "difference" we added in the general case, is that we do not necessarily need the direct common effect to be observed, for the trail to be active, but any descendants may suffice, as they are also an effect, only a bit further down in the chain, but can still be used as evidence.

Now we are ready to state a more powerful property on BNs regarding conditional independence. Assume a BN with the set of variables $\mathcal{X} = \{X_1, \ldots, X_n\}$, and let $\mathbf{Z} \subset \mathcal{X}$ be observed variables. Then if there exists no active trail between two variables $X_i \in \mathcal{X}$ and $X_j \in \mathcal{X}$ given $\mathbf{Z}$, we say that X_i and X_j are *d-separated* given $\mathbf{Z}$. Here the "d" in d-separated means "directional". We say that $\mathbf{Z}$ d-separates X_i from X_j, and this implies that X_i and X_j are conditionally independent given $\mathbf{Z}$.



(a) $(X_5 \perp X_1, X_2, X_4, X_6 \mid X_3)$



(b) $(X_5 \perp X_6 \mid X_3, X_8)$

Figure 2.3: Example showing d-separation in action. Here the variables in blue are conditionally independent of the variable in violet given the variable in orange.

Consider the BN shown in Figure 2.3. To the left, we have that the observation X_3 d-separates X_5 from X_1, X_2, X_4 and X_6, whereas in (b) we have that the observations X_3 and X_8 d-separates X_5 only from X_6. Let us find out why this is the case. In Figure 2.3a, we see the effect of Expression (2.2), since $\text{Pa}(X_5) = \{X_3\}$ and $\text{NonDesc}(X_5) = \{X_1, X_2, X_3, X_4, X_6\}$. However, when we additionally observe X_8 as in Figure 2.3b, we may no longer guarantee conditional independence between X_5 and any of X_1, X_2 or X_4. This is because of the trail $X_5 \to X_7 \leftarrow X_4 \leftarrow X_1 \to X_3 \leftarrow X_2$ suddenly becomes active. This is because X_7 is the middle node in a v-shape of the trail and its child, X_8, is observed. Similarly, X_3 is the middle node in a v-shape of the trail and is observed. There are no other v-shapes in the trail, and no other observed nodes, meaning that the trail is active.

2.2 Dynamic Bayesian networks

DBNs or temporal BNs are networks that that contain the same variables at each time step, where there are additional edges between the variables across adjacent time steps. Figure 2.4 shows a suggested temporal version of the original BN $\mathcal{B}^{\text{sprinkler}}$ in Figure 2.1. Here, a time step has the length of one day, and we have assumed wet grass today may

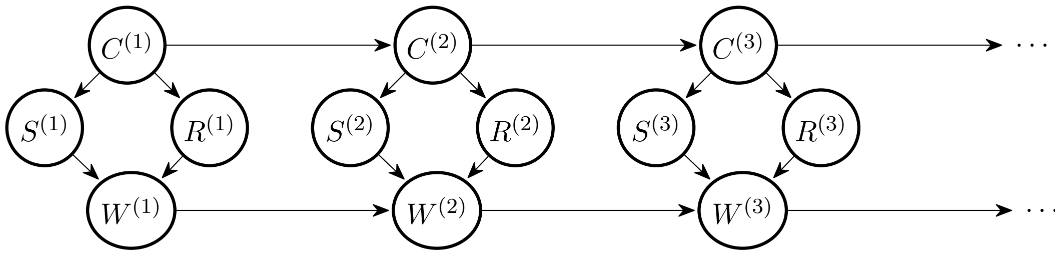


Figure 2.4: DBN, a temporal version of the original BN $\mathcal{B}^{\text{sprinkler}}$ example. The superscript in the variables indicate what day it is, starting at day 1.

cause wet grass tomorrow, so we have the edge $C^{(t)} \to C^{(t+1)}$ for $t = 1, 2, \dots$. Similarly for the cloudy variable $C^{(t)}$, we assume that the sky being cloudy today may cause the sky being cloudy tomorrow. Murphy (2002) provides excellent literature on DBNs, their representations and inference techniques. Other good sources on DBNs include Koller and Friedman (2009, pp. 200–212) and Russell and Norvig (2021, pp. 479–515).

2.2.1 Assumptions

The first assumption we make is that the time can be discretized into time steps $t = 1, 2, \dots$, where the time difference between any two adjacent time steps t and $t+1$ is constant, e.g. 1 second.

We use $\mathcal{X}^{(t)}$ to denote the set of random variables in time step t. The probability distribution over the whole system until some time step T, $\mathcal{P}(\mathcal{X}^{(1)}, \mathcal{X}^{(2)}, \dots, \mathcal{X}^{(T)})$ will often be abbreviated using the shorthand notation $\mathcal{P}(\mathcal{X}^{(1:T)})$. Using the chain rule, we can write this as

$$\mathcal{P}(\mathcal{X}^{(1:T)}) = \mathcal{P}(\mathcal{X}^{(1)}) \prod_{t=1}^{T-1} \mathcal{P}(\mathcal{X}^{(t+1)} \mid \mathcal{X}^{(1:t)}). \tag{2.6}$$

The second assumption we make, the *Markov assumption*, is that knowing precisely the current state of the system, any additional information about the history of the system provides no extra information to infer the state at the next time step. We write this as $(\mathcal{X}^{(t+1)} \perp \mathcal{X}^{(1:(t-1))} \mid \mathcal{X}^{(t)})$, for $t \geq 1$, which says that the state at the next time step $\mathcal{X}^{(t+1)}$, and the history of the system except the current state $\mathcal{X}^{(1:(t-1))}$, are conditionally independent, given the current state $\mathcal{X}^{(t)}$. We call such systems *Markovian.* With this extra assumption, we can simplify Expression (2.6) to

$$\mathcal{P}(\mathcal{X}^{(1:T)}) = \mathcal{P}(\mathcal{X}^{(1)}) \prod_{t=1}^{T-1} \mathcal{P}(\mathcal{X}^{(t+1)} \mid \mathcal{X}^{(t)}). \tag{2.7}$$

The Markov assumption also has the advantage of few parameters, as $\mathcal{P}(\mathcal{X}^{(t+1)} \mid \mathcal{X}^{(t)})$ is often "easier" to formulate than $\mathcal{P}(\mathcal{X}^{(t+1)} \mid \mathcal{X}^{(1:t)})$. For many dynamical systems, this is a reasonable assumption, which vastly reduce complexity of the model. Indeed, if $\mathcal{X}^{(t)} = \{S^{(t)}\}$, where the set of possible outcomes have a cardinality of M states, and we want to know the marginal probability distribution $P(S^{(t)})$, we have to sum over all the possible combinations of the previous states:

$$P(S^{(t)} = s^{(t)}) = \sum_{s^{(1)}} \cdots \sum_{s^{(t-1)}} P(s^{(1:t)}) = \sum_{s^{(1)}} \cdots \sum_{s^{(t-1)}} P(s^{(1)}) \prod_{i=1}^{t-1} P(s^{(i+1)} \mid s^{(1:i)}),$$

for every $s^{(t)}$ giving us a total cost of $\mathcal{O}(M^t)$. For Markovian systems however, the computation can be reduced as follows

$$\begin{aligned}
P(S^{(t)} = s^{(t)}) &= \sum_{s^{(1)}} \cdots \sum_{s^{(t-1)}} P(s^{(1)}) \prod_{i=1}^{t-1} P(s^{(i+1)} \mid s^{(i)}) \\
&= \sum_{s^{(t-1)}} P(s^{(t)} \mid s^{(t-1)}) \cdots \sum_{s^{(2)}} P(s^{(3)} \mid s^{(2)}) \underbrace{\sum_{s^{(1)}} P(s^{(2)} \mid s^{(1)}) P(s^{(1)})}_{M \text{ mult. } +M-1 \text{ summ.}} \\
&= \sum_{s^{(t-1)}} P(s^{(t)} \mid s^{(t-1)}) \ldots \underbrace{\sum_{s^{(2)}} P(s^{(3)} \mid s^{(2)}) P(s^{(2)})}_{M \text{ mult. } +M-1 \text{ summ.}} \\
&= \underbrace{\sum_{s^{(t-1)}} P(s^{(t)} \mid s^{(t-1)}) P(s^{(t-1)})}_{M \text{ multiplications } +M-1 \text{ summations}} ,
\end{aligned}$$

for every $s^{(t)} \in \{1, \ldots, M\}$, giving a total cost of $M(t-1)(2M-1) = \mathcal{O}(M^2 t)$.

The Markov assumption may not always be reasonable, for instance if we are tracking the position of a moving object. Knowing the position of the object at a given time, does not tell us how fast the object is moving. Maybe the object is not moving at all. If we knew the two last positions however, we can estimate its velocity and therefore its next position, so we can relax the Markov assumption to a *second order Markovian system*. This is an improvement, but we cannot tell if the object is increasing in speed, or decreasing, unless we know the *three* last positions of the object, but then we have a third order Markovian system. Another way of getting around this problem, is to let $\mathcal{X}^{(t)} = \{Y^{(t)}, \dot{Y}^{(t)}\}$, where $Y^{(t)}$ is the position of the object and $\dot{Y}^{(t)}$ is the velocity (or even add acceleration, $\ddot{Y}^{(t)}$). With this approach, we can stick to the original Markovian system where we only have a direct dependence on the previous time step.

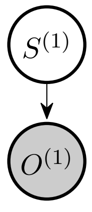
The third assumption we make, the *stationary assumption*, is that it does not matter what t is. The CPD $\mathcal{P}(\mathcal{X}^{(t+1)} \mid \mathcal{X}^{(t)})$ is the same for all $t \geq 1$. This makes sense if we think of the moving object again. That is, knowing the position and velocity of the object at some time $t = t^*$, the probability of the object having a certain position at time $t+1$ is independent of what t^* is, if we are tracking e.g. the position of a person riding down a water slide. If we are tracking the position of a car, however, it might matter what t is, since at certain hours it his more likely to be rush hour. A workaround, is to add a variable $R^{(t)}$ to $\mathcal{X}^{(t)}$, to tell the model whether there is rush hour or not, and proceed with the stationarity assumption. Since the time t does not matter, we often use the notation $\mathcal{P}(\mathcal{X}' \mid \mathcal{X})$ to denote what we call the *transition model* $\mathcal{P}(\mathcal{X}^{(t+1)} \mid \mathcal{X}^{(t)})$, $t \geq 1$.

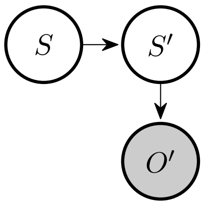
2.2.2 Definition

With the three basic assumptions, we can, with only a few parameters construct a BN which keeps on growing as time passes. The Markov assumption reduces the number of parameters needed for the model since $P(\mathcal{X}^{(t+1)} \mid \mathcal{X}^{(1:t)}) = P(\mathcal{X}^{(t+1)} \mid \mathcal{X}^{(t)})$, and the stationarity assumption further reduces the number, since t does not matter for the CPD $P(\mathcal{X}^{(t+1)} \mid \mathcal{X}^{(t)})$.

A DBN is a pair $\langle \mathcal{B}_1, \mathcal{B}_\rightarrow \rangle$ over the variables $\mathcal{X}^{(1)}, \mathcal{X}^{(2)}, \ldots$, where $\mathcal{B}_1$ is the BN over the variables in the first time step $\mathcal{X}^{(t)}$, and $\mathcal{B}_\rightarrow$ is a *conditional* BN over the variables $\mathcal{X}^{(t+1)}$ given the variables $\mathcal{X}_I^{(t)} = \{X^{(t)} : X^{(t)} \in \mathcal{X}^{(t)} \cap \mathrm{Pa}(\mathcal{X}^{(t+1)})\}$ for $t = 1, 2, \ldots$. That is, the distribution over $\mathcal{B}_\rightarrow$ is $\mathcal{P}(\mathcal{X}' \mid \mathcal{X}_I)$. The variables $\mathcal{X}_I$ are called the *interface variables* (Murphy, 2002; Koller & Friedman, 2009), as they are the variables in the current time step that directly influences variables in the next time step. Since the transition model $P(\mathcal{X}' \mid \mathcal{X}_I)$ only needs information from two adjacent time-steps, the BN $\mathcal{B}_\rightarrow$ is commonly called a *2-time step Bayesian network* (2-TBN) (Murphy, 2002; Koller & Friedman, 2009).

The most simple example of a DBN, is a HMM, where $\mathcal{X}^{(t)} = \{S^{(t)}, O^{(t)}\}$ as shown in Figure 2.5. Here, $S^{(t)}$ is the hidden variable that we are interested in, and $O^{(t)}$ is an observed variable that depends on $S^{(t)}$. Figure 2.5a and 2.5b show the components needed to define the whole network, where Figure 2.5c shows how we use the components to unroll DBN, starting with $\mathcal{B}_1$ and add $\mathcal{B}_\rightarrow$ repeatedly. Here, we see that the interface


(a) HMM $\mathcal{B}_1$.


(b) HMM $\mathcal{B}_\rightarrow$.

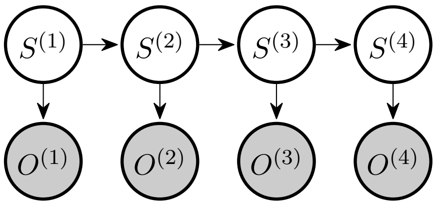
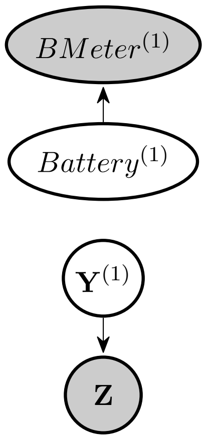

(c) HMM unrolled for four time steps.

Figure 2.5: HMMs represented in two ways. (a) and (b) shows the minimal representation $\langle \mathcal{B}_1, \mathcal{B}_\rightarrow \rangle$, while (c) shows the complete network unrolled for four time steps. White nodes are hidden and shaded/gray nodes are observed.

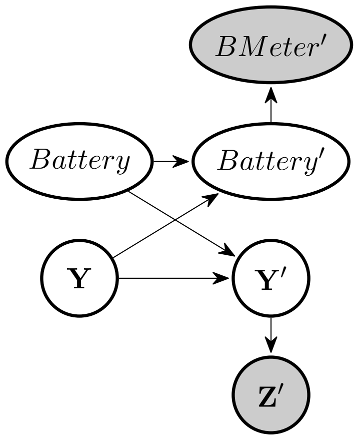
variable is $\mathcal{X}_I = \{S\}$, since $S \rightarrow S'$ is the only edge going from one time step to the next.

Another example adapted from Russell and Norvig (2021) is shown in Figure 2.6, which models the motion of a robot in the X-Y plane. The hidden variables are $Battery^{(t)}$

which is the true battery level at time t, and $\mathbf{Y}^{(t)}$ is a vector containing information about the position and velocity of the robot. The observed variables are $BMeter^{(t)}$ which is the estimated battery level and $\mathbf{Z}^{(t)}$ which is the observed position collected from GPS data.



(a) $\mathcal{B}_1$.



(b) $\mathcal{B}_{\rightarrow}$.

Figure 2.6: Another DBN adapted from Russell and Norvig (2021), represented in two ways. (a) and (b) shows the minimal representation $\langle \mathcal{B}_1, \mathcal{B}_{\rightarrow} \rangle$.

Finally, we show in Figure 2.7 the 2-TBN of the temporal version of the sprinkler example from Figure 2.4. Here, we have also assumed that we only observe if the grass is wet, whereas the other variables are hidden.

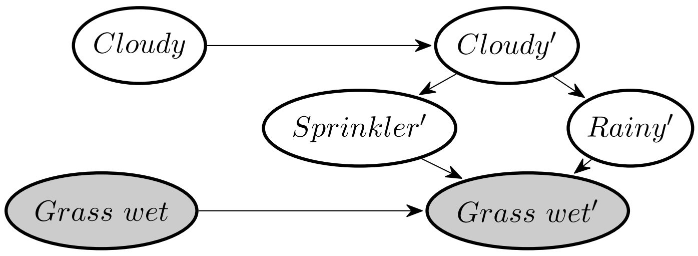


Figure 2.7: 2-TBN of the temporal version of the sprinkler example.

2.3 State-observation models

When introducing the DBNs, we saw that some of the variables were observed, while others were hidden. There is a very special class of DBNs that we are particularly interested in, and will be the topic for the rest of the thesis. That is, when we are tracking a dynamical process where we do not know its true state, we can do indirect observations to estimate its state. This is called the *state-observation model*. Here, we can think of

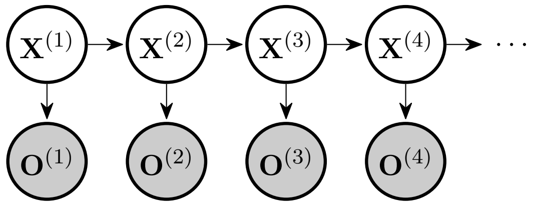


Figure 2.8: State-observation model representation. Here, $\mathbf{X}^{(t)}$ is a set of hidden variables and $\mathbf{O}^{(t)}$ is a set of observed variables.

the DBN as a combination of two processes. The variables are $\mathcal{X}^{(t)} = \mathbf{X}^{(t)} \cup \mathbf{O}^{(t)}$, where $\mathbf{X}^{(t)}$ and $\mathbf{O}^{(t)}$ are disjoint sets. The variables in $\mathbf{X}^{(t)}$ are part the dynamical system that evolves on its own, while the variables in $\mathbf{O}^{(t)}$ are part of the observation process, that depends on $\mathbf{X}^{(t)}$. That is, the hidden variables $\mathbf{X}^{(t)}$ contain all the interface variables $\mathcal{X}_I^{(t)}$. The transition model is $\mathcal{P}(\mathbf{X}' \mid \mathbf{X})$ and the observation model is $\mathcal{P}(\mathbf{O} \mid \mathbf{X})$.

The graphical representation of state-observation models is shown in Figure 2.8. This looks almost identical to the HMM, so it may come as no surprise to reveal that the HMM is a state-observation model. Out of the three DBN examples we have shown this far, two of them can be represented as a state-observation model. The HMM is trivial, where the hidden variable is $\mathbf{X}^{(t)} = \{S^{(t)}\}$ and the observed variable is $\mathbf{O}^{(t)} = \{O^{(t)}\}$. The robot example in Figure 2.6 can also be classified as a state-observation model where the hidden variables are $\mathbf{X}^{(t)} = \{Battery^{(t)}, \mathbf{Y}^{(t)}\}$ and the observed variables are $\mathbf{O}^{(t)} = \{BMeter^{(t)}, \mathbf{Z}^{(t)}\}$. We see that the observed variables depend only on variables within the same time step, and $\mathbf{X}^{(t)} = \mathcal{X}_I^{(t)}$.

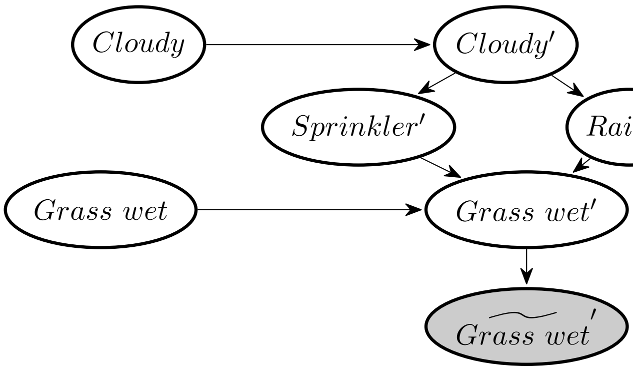


Figure 2.9: The sprinkler example rerepresented as a state-observation model, where we split the $Wet\ grass$ variable in two separate variables.

In the sprinkler example in Figure 2.7 however, we have a problem with the $Wet\ grass$ variable. It is both an interface variable, and an observed variable. This DBN would then not be classified as a state-observation model. This can easily be solved however. In fact, any DBN can be represented as a state-observation model. Indeed, if we introduce a new variable $\widetilde{Wet\ grass} \in \mathbf{O}$, which is the observation that the grass is wet, and the original $Wet\ grass \in \mathbf{X}$ is hidden and part of the dynamical process. See Figure 2.9. We let them be deterministically equal, that is $P(\widetilde{Grass\ wet} \mid Grass\ wet) = \mathbb{I}\{\widetilde{Grass\ wet} = Grass\ wet\}$, where $\mathbb{I}\{A\}$ is the indicator function that evaluates to 0 if the event A is false, and 1 if A is true. Representing the DBN as a state-observation model is convenient so we can apply the same inference techniques which require this form.

2.3.1 HMMs

As mentioned, the simplest form of a state-observation model is the HMM, where each time step only consist of two variables: $\mathbf{X}^{(t)} = \{S^{(t)}\}$ and $\mathbf{O}^{(t)} = \{O^{(t)}\}$, see Figure 2.5. Here S is discrete, and O is usually discrete too, but can be continuous, usually Gaussian distributed, given the S outcome (Scott, 2002; Murphy, 2002; Zucchini & MacDonald, 2009). Assuming discrete O, the transition model and observation model can be represented by CPTs. Often, it is convenient to write the CPTs as vectors and matrices so we can take advantage of the operations in linear algebra when we perform inference. We have

$$\begin{aligned}
P(S^{(1)}) &= \boldsymbol{\pi} = (\pi_i) \in \mathbb{R}^{M} && \text{(Initial distribution)}\\
P(S' \mid S) &= \mathbf{P} = (p_{ij}) \in \mathbb{R}^{M\times M} && \text{(Transition model)}\\
P(O \mid S) &= \mathbf{E} = (e_{ij}) \in \mathbb{R}^{M\times L} && \text{(Observation model)},
\end{aligned} \tag{2.8}$$

where $\pi_i = P(S^{(1)} = i)$, $p_{ij} = P(S' = j \mid S = i)$ and $e_{ij} = P(O = j \mid S = i)$. The matrix $\mathbf{P}$ is called the *transition probability matrix*, and $\mathbf{E}$ is called the *observation probability matrix* (or *emission probability matrix*).

Consider a HMM with $S \in \{1, 2, 3, 4\}$ and $O \in \{1, 2\}$ with the following parameters:

$$\begin{aligned}
P(S^{(1)}) = \boldsymbol{\pi} &= \begin{bmatrix} 0.9 & 0 & 0.1 & 0 \end{bmatrix}^{\mathsf{T}};\\
P(S' \mid S) = \mathbf{P} &= \begin{bmatrix} 0.8 & 0.2 & 0 & 0\\ 0 & 0.7 & 0.3 & 0\\ 0 & 0 & 0.5 & 0.5\\ 0.9 & 0 & 0 & 0.1 \end{bmatrix};\\
P(O \mid S) = \mathbf{E} &= \begin{bmatrix} 0.2 & 0.8\\ 0.3 & 0.7\\ 0.9 & 0.1\\ 0.6 & 0.4 \end{bmatrix},
\end{aligned} \tag{2.9}$$

As with the CPTs, we see that the rows sum to one, and the elements in the initial distribution also sums to one.

If the probability matrices are sparse, it may be more convenient to represent them as a graph, a different graph not to be confused with BNs. Here, the nodes correspond to the different values the state variable and observation variable can take, and where the edges represent the probabilities of the transition process and observation process.

If there is no edge going from one state to another, it means that the probability is zero of making that transition. Figure 2.10 shows such a graphical representation of Expression (2.9). As we see, there are no edges going from e.g. $S = 2$ to $S = 1$, meaning

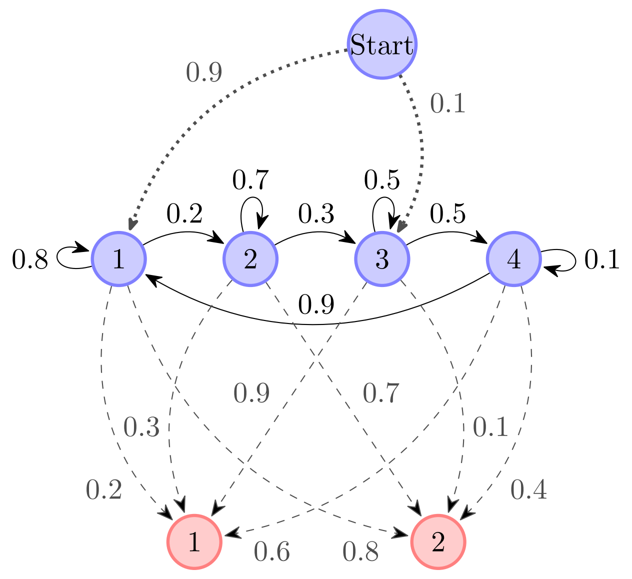


Figure 2.10: A complete graphical representation of an HMM. The node at the top which is labeled "Start", indicates the start of the process, where the dotted edges represents the initial probabilities. The blue nodes labeled $1, 2, 3, 4$ represent the different states of the state variable, and the solid edges represent the transition probabilities. The red nodes labeled 1 and 2 represent the observed values where the dashed edges represent the observation probabilities. The model is mathematically described in Expression (2.9).

that there is a zero probability of transitioning from 2 to 1 directly. It actually has to transition via 3 and 4 before it can reach 1.

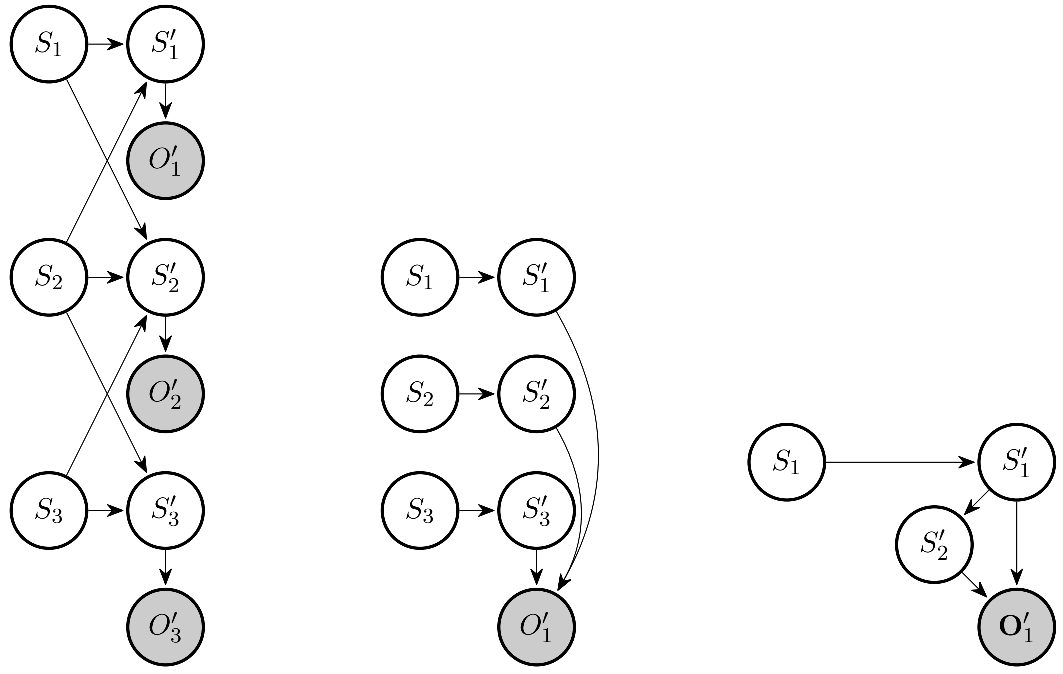
As mentioned, the observed variable can also be continuous, typically Gaussian. Then the observation model has the form

$$O \mid \{S = s\} \sim \mathcal{N}(\mu_s, \sigma_s^2), \quad s = 1, \ldots, M.$$

That is, the mean and variance may change depending on S. We will come back to hybrid models in Section 2.3.3. More complex HMMs exist. Some common HMMs that are also state-observation models are shown in Figure 2.11.

2.3.2 LDSs

LDSs, also called *state space models* can be thought of as the continuous equivalent of HMMs (West & Harrison, 2006; Koller & Friedman, 2009; Särkkä, 2013). The hidden variable is a continuous N-vector $\mathbf{Y}$, and the observed variable is a continuous L-vector $\mathbf{O}$. The LDS is shown in Figure 2.12, where we see that the structure is identical to HMMs and the general form of state-observation models.



(a) Coupled HMM with three chains

(b) Factorial HMM with three chains

(c) HMM with mixture of Gaussian observations.

Figure 2.11: 2-TBN of more general HMMs (Murphy, 2002).

The transition model and the observation model are *conditional linear Gaussians* (CLGs). That is, they have the following form:

$$\begin{aligned}
\mathbf{Y}^{(1)} &\sim \mathcal{N}(\boldsymbol{\nu}, \boldsymbol{\Gamma}) && \text{(Initial distribution)} \\
\mathbf{Y}' \mid \{\mathbf{Y} = \mathbf{y}\} &\sim \mathcal{N}(\mathbf{A}\mathbf{y} + \mathbf{b}, \mathbf{Q}) && \text{(Transition model)} \\
\mathbf{O} \mid \{\mathbf{Y} = \mathbf{y}\} &\sim \mathcal{N}(\mathbf{C}\mathbf{y} + \mathbf{d}, \mathbf{R}) && \text{(Observation model),}
\end{aligned} \tag{2.10}$$

where $\boldsymbol{\Gamma}$ and $\mathbf{Q}$ are $N \times N$ (symmetric non-negative definite) covariance matrices, $\mathbf{A}$ an $N \times N$ matrix, $\boldsymbol{\nu}$ and $\mathbf{b}$ are length-N vectors, $\mathbf{C}$ is an $L \times N$ matrix, $\mathbf{R}$ is an $L \times L$ covariance matrix and $\mathbf{d}$ is a length-L vector. The name *linear* comes from exactly that

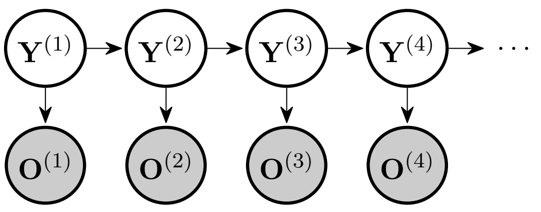


Figure 2.12: LDS, where $\mathbf{Y}^{(t)}$ is the dynamic hidden state, and $\mathbf{O}^{(t)}$ is the observed state.

each entry in $\mathbf{Y}'$ is a linear function of the entries in the previous state $\mathbf{Y} = \mathbf{y}$, i.e.

$$\mathbf{Y}_i^{(t)} = a_{i1}y_1^{(t)} + a_{i2}y_2^{(t)} + \cdots + a_{iN}y_N^{(t)} + b_i + \epsilon_i^{(t+1)}, \quad \epsilon_i^{(t+1)} \sim \mathcal{N}(0, q_{ii}), \quad i = 1, \ldots, N,$$

where a_{ij} and q_{ij} are the entries in the ith row and jth column of matrices $\mathbf{A}$ and $\mathbf{Q}$, respectively. Figure 2.13 shows an example of a CLG in the univariate case. The

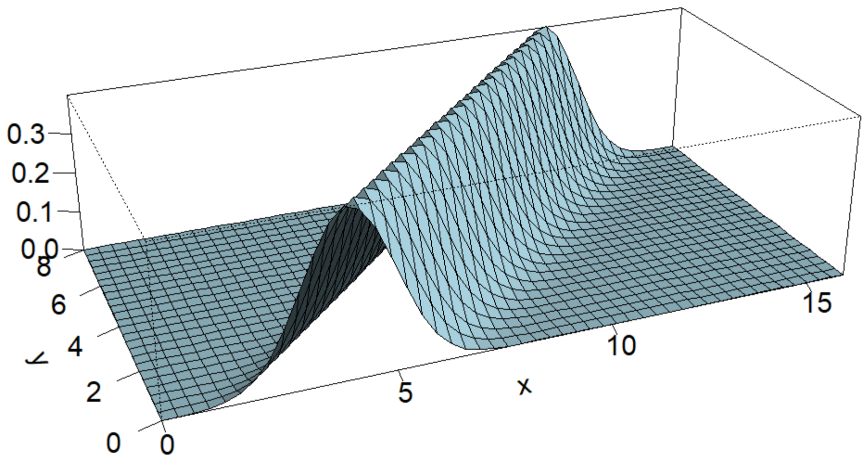


Figure 2.13: PDF of a CLG where $p_X(x \mid y) \sim \mathcal{N}(y + 4, 1)$.

advantage of parameterizing LDSs as CLGs is that no matter what variables in the network we condition on, and no matter what variables we compute the PDF of, the result is always a CLG, which we show in Chapter 3.3.

2.3.3 SLDSs

SLDSs extends LDSs to include a discrete switching state $S^{(t)} = 1, 2, \ldots, M$, where for each possible outcome, we have a distinct LDS. That is the hidden variables are $\mathbf{X}^{(t)} = \{S^{(t)}, \mathbf{Y}^{(t)}\}$, where $\mathbf{Y}^{(t)}$ is an N-vector, and the observed variable is the L-vector $\mathbf{O}^{(t)}$.

Figure 2.14 shows the graphical structure of the network. That is, the switching state S' depends only on the previous switching state S, where as the dynamic state $\mathbf{Y}'$ depends on the current switching state S' as well as the previous dynamic state $\mathbf{Y}$. The observed state $\mathbf{O}$ depends on the dynamic state $\mathbf{Y}$ and optionally, the switching state S in the same time step.

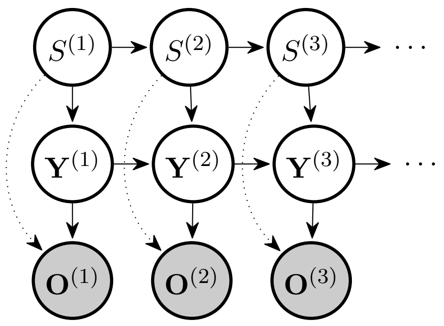


Figure 2.14: SLDS, where $S^{(t)}$ is the (discrete) switching state, $\mathbf{Y}^{(t)}$ is the (continuous) dynamic state, and $\mathbf{O}^{(t)}$ is the (continuous) observed state. Dotted edge means optional dependence.

The initial, transition and observation model are given by

$$\begin{aligned}
P(S^{(1)}) &= \boldsymbol{\pi} = (\pi_i) \in \mathbb{R}^M && \text{(Initial distribution)} \\
\mathbf{Y}^{(1)} \mid \{S^{(1)} = s\} &\sim \mathcal{N}(\boldsymbol{\nu}_s, \boldsymbol{\Gamma}_s) && \text{(Initial distribution)} \\
P(S' \mid S) &= \mathbf{P} = (p_{ij}) \in \mathbb{R}^{M \times M} && \text{(Transition model)} \\
\mathbf{Y}' \mid \{S' = s', \mathbf{Y} = \mathbf{y}\} &\sim \mathcal{N}(\mathbf{A}_{s'}\mathbf{y} + \mathbf{b}_{s'}, \mathbf{Q}_{s'}) && \text{(Transition model)} \\
\mathbf{O} \mid \{S = s, \mathbf{Y} = \mathbf{y}\} &\sim \mathcal{N}(\mathbf{C}_s\mathbf{y} + \mathbf{d}_s, \mathbf{R}_s) && \text{(Observation model).}
\end{aligned} \tag{2.11}$$

Notice that if $S^{(t)}$ has size $M = 1$, then the whole network collapses to a simple LDS. Generally, the size of $S^{(t)}$ determines how many LDSs we are working with and their possible interactions can be determined from the transition probability matrix $\mathbf{P}$. This way we can model a nonlinear dynamic behavior as piece-wise linear behavior, where we assume that only one type of linear behavior happens at the time, such as the motion of an aircraft (climbing, descending, maneuver).

Cases where the observation variable depends on the switching state could be when the switching state includes an indicator variable which tells whether or not the sensor we are receiving observations from is broken. For the rest of the thesis, we omit this optional edge such that the observation model takes the form

$$\mathbf{O} \mid \{\mathbf{Y} = \mathbf{y}\} \sim \mathcal{N}(\mathbf{C}\mathbf{y} + \mathbf{d}, \mathbf{R}).$$

Chapter 3

Methods

There are three main inference task in state-observation models, *filtering*, *smoothing* and prediction, see Figure 3.1. It is called filtering when for each time step we find the probability distribution over the hidden variables given the observations up until that time step, and it is called smoothing when for each time step we find the probability distribution over the hidden variables given *all* observations. That is, we combine our prior knowledge about the hidden states together with the observations to get a much better estimate of what the hidden states might be at each time step. Our focus is filtering, as this is excellent for monitoring or tracking something live, where we receive data in real time. For offline analysis where the all the data already has been collected, smoothing may be of interest.

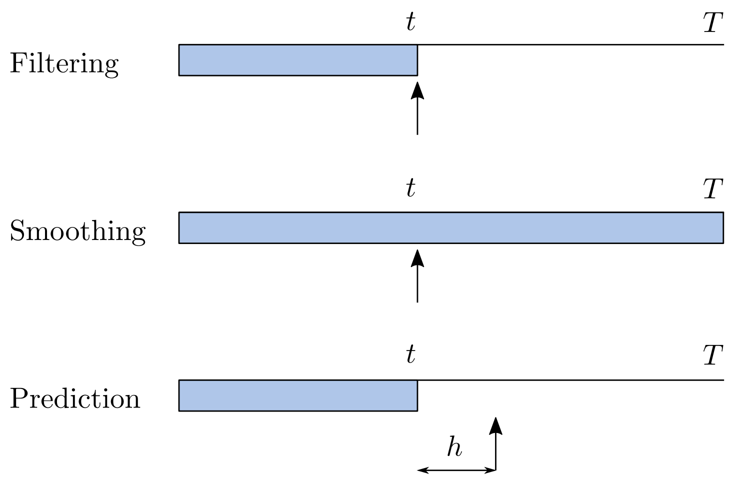


Figure 3.1: The three main types of inference that are performed on state-observation models. The arrow indicate at what time step we perform inference on, and the blue area indicate where we have data. T is the length of the sequence.

The term "filtering" comes from when we in the process of obtaining the "best estimate" from noisy data, we are "filtering out" the noise (Bar-Shalom et al., 2004).

For instance, when we are trying to hear what is being said on a radio, but due to bad signal, the sound that comes out is literally noisy. Then by combining our prior knowledge about what is being said, i.e. we know the context, together with what we hear, we are in our mind "filtering" out the noise in the radio and get more information. The term "smoothing" comes from when after we first have done the filtering and go backwards recursively to condition on all observations, the distribution usually becomes much smoother, because of the additional data at later time steps.

Think of a scenario where we are monitoring whether or not a machine part malfunctions. When we perform filtering, we carefully consider each observation, as it may be the one that shows a malfunction. This makes the filtered estimate to look a bit noisy. When the machine part actually malfunctions, we detect some "change" in the observations, and the probability of a malfunction increases, but as this may also be considered as just noisy data, this increase can be slow and noisy until more observations come in and we see that the "change" is persistent. When this is concluded, it might be interesting to know at what time it broke. This is where smoothing helps, as we can condition on later observations, and since we now know that the machine part malfunctioned, it is easier to detect when it most likely happened, and thus it "smooths" out the probability distribution.

The *filtering algorithm* (Pearl, 1988; Murphy, 2002; Lerner, 2003; Bar-Shalom et al., 2004; Koller & Friedman, 2009; Särkkä, 2013) is a recursive algorithm, where we by knowing the filtered estimate at some time t, we can use this estimate together with the next incoming observational data at $t+1$ to compute the filtered estimate at the next time step $t+1$.

In Section 3.1 we first show the general approach to the filtering algorithm, then we show specific examples for HMMs in Section 3.2 and for the KF for LDSs in Section 3.3. Then we come to the more difficult case for SLDSs in Section 3.4. First we show how to do exact filtering for SLDSs, before we explore some possible approximations and compare them. Then in Section 3.5 we show the extended Kalman filter (EKF) (Bar-Shalom et al., 2004) which is used when the dynamical process must be modelled as a nonlinear process.

3.1 General filtering algorithm

We have the state-observation model $\mathcal{X}^{(t)} = \mathbf{X}^{(t)} \cup \mathbf{O}^{(t)}$, $t = 1, 2, \ldots$, where the hidden variables $\mathbf{X}^{(t)}$ and the observed variables $\mathbf{O}^{(t)}$ may contain both discrete and continuous variables. We recall that the state-observation model is given by the transition model $\mathcal{P}(\mathbf{X}' \mid \mathbf{X})$ with initial distribution $\mathcal{P}(\mathbf{X}^{(1)})$, and the observation process $\mathcal{P}(\mathbf{O} \mid \mathbf{X})$. The *filtered estimate* is defined as

$$\sigma^{(t)}(\mathbf{x}^{(t)}) = \mathcal{P}(\mathbf{X}^{(t)} = \mathbf{x}^{(t)} \mid \mathbf{o}^{(1:t)}), \quad t = 1, 2, \ldots. \tag{3.1}$$

That is, we condition on the observation sequence $\mathbf{O}^{(1)} = \mathbf{o}^{(1)}, \ldots, \mathbf{O}^{(t)} = \mathbf{o}^{(t)}$, to get a more precise estimate of the hidden state(s) $\mathbf{X}^{(t)}$.

For now, when we compare the computational complexities, we assume that $\mathcal{X}$ contain discrete variables only. The naïve approach to compute the filtered estimate, is to straightforwardly apply Bayes' rule and the law of total probability. We use the shorthand notation $\sum_{\mathbf{x}^{(1:t)}}$ to mean the sum over all possible combinations of $\mathbf{X}^{(1)}, \ldots, \mathbf{X}^{(t)}$ in the sample space of $\mathbf{X}^{(t)}$.

$$\begin{aligned}\sigma^{(t)}(\mathbf{x}^{(t)}) \propto \mathcal{P}(\mathbf{x}^{(t)}, \mathbf{o}^{(1:t)}) &= \sum_{\mathbf{x}^{(1:(t-1))}} \mathcal{P}(\mathbf{x}^{(1:t)}, \mathbf{o}^{(1:t)}) \\ &= \sum_{\mathbf{x}^{(1:(t-1))}} \mathcal{P}(\mathbf{x}^{(1)}) P(\mathbf{o}^{(1)} \mid \mathbf{x}^{(1)}) \prod_{t'=1}^{t-1} \mathcal{P}(\mathbf{x}^{(t'+1)} \mid \mathbf{x}^{(t')}) \mathcal{P}(\mathbf{o}^{(t'+1)} \mid \mathbf{x}^{(t'+1)}),\end{aligned} \tag{3.2}$$

where we in the the last equality applied the Bayesian chain rule. Assuming $\mathbf{X}$ can take M possible unique assignments, then for each $\mathbf{x}^{(t)}$ we have M^{t-1} summations, and for every summation we have $2t-1$ multiplications, which results in a demanding complexity of $\mathcal{O}(tM^t)$ operations.

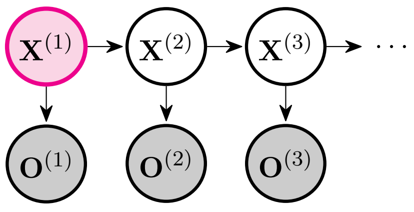
This motivates the search for an alternative approach that does not grow exponentially, and lucky for us, such an approach exists, and is very efficient both in computing and requires less memory. This is a recursive approach where we make use of the stationarity assumption and Markov assumption described in Section 2.2.1, in that you only need the result from the previous time step to compute the next result. That is, we assume we already know $\sigma^{(t)}(\mathbf{x}^{(t)})$, and want to compute $\sigma^{(t+1)}(\mathbf{x}^{(t+1)})$. We break it down into two steps: the *prediction step*, and the *condition step* (or *update step*). In the prediction step, we compute the probability distribution over $\mathbf{x}^{(t+1)}$ given the same observation sequence as for $\sigma^{(t)}(\mathbf{x}^{(t)})$:

$$\sigma^{(\cdot t+1)}(\mathbf{x}) = \mathcal{P}(\mathbf{X}^{(t+1)} = \mathbf{x} \mid \mathbf{o}^{(1:t)}). \tag{3.3}$$

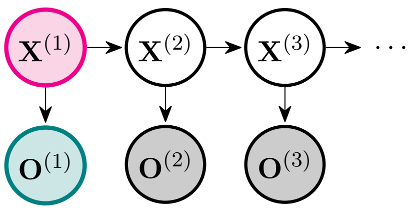
We call this the *predicted estimate.* Note that we added a *dot* in the superscript $(\cdot t+1)$, to specify that we are conditioning on the observations up to time t. In the condition step, we include the observation at time $t+1$ to get the filtered estimate at time $t+1$, i.e. $\sigma^{(t+1)}(\mathbf{x})$. A schematic of the procedure is shown in Figure 3.2. We walk through the steps in detail below.

Assume we know $\sigma^{(t)}(\mathbf{x}^{(t)})$. Then

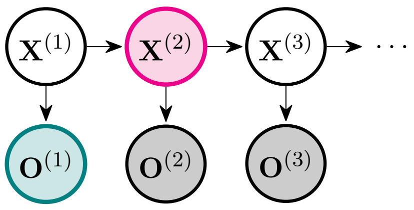
$$\begin{aligned}\sigma^{(\cdot t+1)}(\mathbf{x}^{(t+1)}) &= \mathcal{P}(\mathbf{x}^{(t+1)} \mid \mathbf{o}^{(1:t)}) \\ &= \sum_{\mathbf{x}^{(t)}} \mathcal{P}(\mathbf{x}^{(t)}, \mathbf{x}^{(t+1)} \mid \mathbf{o}^{(1:t)}) \\ &= \sum_{\mathbf{x}^{(t)}} \mathcal{P}(\mathbf{x}^{(t+1)} \mid \mathbf{x}^{(t)}, \mathbf{o}^{(1:t)}) \mathcal{P}(\mathbf{x}^{(t)} \mid \mathbf{o}^{(1:t)}) \\ &= \sum_{\mathbf{x}^{(t)}} \mathcal{P}(\mathbf{x}^{(t+1)} \mid \mathbf{x}^{(t)}) \sigma^{(t)}(\mathbf{x}^{(t)}) \\ &= \sum_{\mathbf{x}^{(t)}} \mathcal{P}(\mathbf{X}' = \mathbf{x}^{(t+1)} \mid \mathbf{X} = \mathbf{x}^{(t)}) \sigma^{(t)}(\mathbf{x}^{(t)}),\end{aligned} \tag{3.4}$$



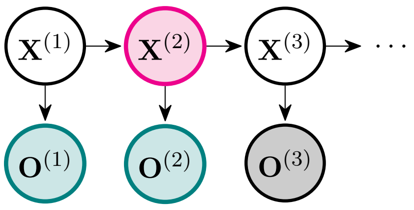
(a) Base case, $\mathcal{P}(\mathbf{X}^{(1)} = \mathbf{x})$.



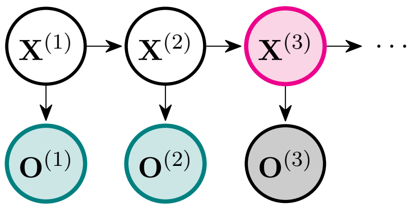
(b) Condition step, $\sigma^{(1)}(\mathbf{x})$.



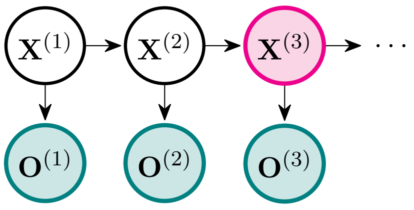
(c) Prediction step, $\sigma^{(\cdot 2)}(\mathbf{x})$.



(d) Condition step, $\sigma^{(2)}(\mathbf{x})$.

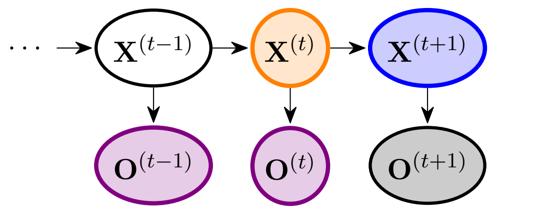


(e) Prediction step, $\sigma^{(\cdot 3)}(\mathbf{x})$.

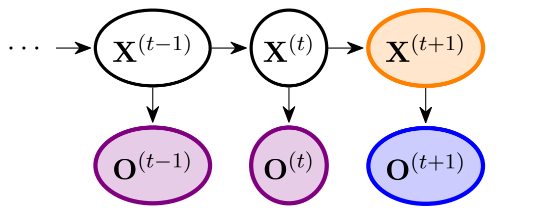


(f) Condition step, $\sigma^{(3)}(\mathbf{x})$.

Figure 3.2: The forward algorithm. The current variable that we compute the distribution over is colored in magenta, and the variables we condition on are colored in teal (blue/green).



(a) Conditional independence used in the prediction step, $(\mathbf{X}^{(t+1)} \perp \mathbf{O}^{(1:t)} \mid \mathbf{X}^{(t)})$.



(b) Conditional independence used in the condition step, $(\mathbf{O}^{(t+1)} \perp \mathbf{O}^{(1:t)} \mid \mathbf{X}^{(t+1)})$.

Figure 3.3: D-separation in the graph used in the forward algorithm.

where we are left with a sum over the product between the transition model $\mathcal{P}(\mathbf{X}' \mid \mathbf{X})$, and the filtered estimate of the previous time step $\sigma^{(t)}(\cdot)$. In the fourth equality we used the BN property of conditional independence between $\mathbf{X}^{(t+1)}$ and $\mathbf{O}^{(1:t)}$ given $\mathbf{X}^{(t)}$, which is shown in Figure 3.3a. If there are continuous variables in $\mathbf{X}^{(t)}$, the form is the same, with the exception that we replace the summation with integration. The prediction step ends up with $2M-1$ operations for every $\mathbf{x}^{(t+1)}$.

Now we condition on the next observation $\mathbf{o}^{(t+1)}$. In the following computation, it is useful to split the observation sequence $\mathbf{o}^{(1:(t+1))}$ into two sets $\mathbf{o}^{(1:t)}$ and $\mathbf{o}^{(t+1)}$:

$$\begin{aligned}
\sigma^{(t+1)}(\mathbf{x}^{(t+1)}) &= \mathcal{P}(\mathbf{x}^{(t+1)} \mid \mathbf{o}^{(1:t)}, \mathbf{o}^{(t+1)}) \\
&\propto \mathcal{P}(\mathbf{x}^{(t+1)}, \mathbf{o}^{(t+1)} \mid \mathbf{o}^{(1:t)}) \\
&= \mathcal{P}(\mathbf{o}^{(t+1)} \mid \mathbf{x}^{(t+1)}, \mathbf{o}^{(1:t)})\mathcal{P}(\mathbf{x}^{(t+1)} \mid \mathbf{o}^{(1:t)}) \\
&= \mathcal{P}(\mathbf{o}^{(t+1)} \mid \mathbf{x}^{(t+1)})\sigma^{(\cdot t+1)}(\mathbf{x}^{(t+1)}) \\
&= \mathcal{P}(\mathbf{O} = \mathbf{o}^{(t+1)} \mid \mathbf{X} = \mathbf{x}^{(t+1)})\sigma^{(\cdot t+1)}(\mathbf{x}^{(t+1)}),
\end{aligned} \tag{3.5}$$

which is simply the product of the observation model $\mathcal{P}(\mathbf{O} \mid \mathbf{X})$ and the prior belief state $\sigma^{(\cdot t+1)}(\cdot)$. Again, for continuous variables in $\mathbf{X}^{(t)}$, we integrate instead. The normalizing constant is found by summing over $\mathbf{x}^{(t)}$,

$$\mathcal{P}(\mathbf{o}^{(t+1)} \mid \mathbf{o}^{(1:t)}) = \sum_{\mathbf{x}} \mathcal{P}(\mathbf{O} = \mathbf{o}^{(t+1)} \mid \mathbf{X} = \mathbf{x})\sigma^{(\cdot t+1)}(\mathbf{x}). \tag{3.6}$$

In the fourth step, we again used the conditional independence property of the BN, which is shown in Figure 3.3b. The condition step takes M operations (because of the normalization step) for every $\mathbf{x}^{(t)}$.

With the base case $\sigma^{(\cdot 1)}(\mathbf{x}) = \mathcal{P}(\mathbf{X}^{(1)} = \mathbf{x})$, computing $\sigma^{(t)}(\mathbf{x})$ now takes $M^2 + Mt((2M-1)+M) = \mathcal{O}(tM^2)$ operations. Compare this to the naïve approach in Expression (3.2), which takes $\mathcal{O}(tM^t)$ operations. This shows us the computational advantage we get with BNs by taking advantage of their conditional independencies! Also, computing $\sigma^{(t)}(\mathbf{x})$ naturally includes all the previous filtered estimates $\sigma^{(1)}(\mathbf{x}), \ldots, \sigma^{(t-1)}(\mathbf{x})$, since they are computed as intermediate steps. What is brilliant with this recursive approach, is that when we use this forward algorithm in a live setting, that is, we are updating the filtered estimates as we obtain data, we only need to keep the current filtered estimate in memory to compute the next filtered estimate. This means that we can discard older filtered estimates to free up memory without losing information about future states.

3.2 Discrete - HMM

Recall the HMM with the state variable S and observation variable O shown in Figure 2.5. Now, S and O are discrete random variables where the cardinality of the possible outcomes are M and L, respectively. The full model description is given by Expression (2.8), where the model parameters are $\boldsymbol{\pi}$, $\mathbf{P}$ and $\mathbf{E}$.

Here the filtered estimate is given by

$$\sigma^{(t)}(s) = P(S^{(t)} = s \mid o^{(1:t)}), \quad s = 1, \dots, M.$$

The prediction step $\sigma^{(\cdot t+1)}(\cdot)$ is computed as

$$\begin{aligned}\sigma^{(\cdot t+1)}(s) &= \sum_{j=1}^{M} P(S' = s \mid S = j)\sigma^{(t)}(j) \\ &= \sum_{j=1}^{M} p_{js}\sigma^{(t)}(j), \quad s = 1, \dots, M,\end{aligned}$$

which we can rewrite as the simple matrix vector product

$$\boldsymbol{\sigma}^{(\cdot t+1)} = \mathbf{P}\boldsymbol{\sigma}^{(t)}, \qquad \text{(Prediction step)} \qquad (3.7)$$

where $\boldsymbol{\sigma}^{(t)} = (\sigma^{(t)}(1), \dots, \sigma^{(t)}(M))$. The condition step $\sigma^{(t+1)}(\cdot)$ is computed as follows:

$$\begin{aligned}\widetilde{\sigma}^{(t+1)}(s) &= P(O = o^{(t+1)} \mid S = s)\sigma^{(\cdot t+1)}(s) \\ &= e_{s,o^{(t+1)}}\sigma^{(\cdot t+1)}(s); \\ \Rightarrow \sigma^{(t+1)}(s) &= \frac{1}{\sum_{i=1}^{M} \widetilde{\sigma}^{(t+1)}(i)}\widetilde{\sigma}^{(t+1)}(s), \quad s = 1, \dots, M\end{aligned}$$

which we can rewrite as

$$\begin{aligned}\widetilde{\boldsymbol{\sigma}}^{(t+1)} &= \mathbf{E}_{:,o^{(t+1)}} \odot \boldsymbol{\sigma}^{(\cdot t+1)}; \\ \Rightarrow \boldsymbol{\sigma}^{(t+1)} &= \frac{1}{\sum_{i=1}^{M} \widetilde{\sigma}^{(t+1)}(i)}\widetilde{\boldsymbol{\sigma}}^{(t+1)}.\end{aligned} \qquad \text{(Condition step)} \qquad (3.8)$$

Here $\mathbf{E}_{:,j}$, means the jth column vector of matrix $\mathbf{E}$, and $\odot$ is the element-wise product operator. The base case is simply

$$\boldsymbol{\sigma}^{(\cdot 1)} = \boldsymbol{\pi}. \qquad \text{(Base case)} \qquad (3.9)$$

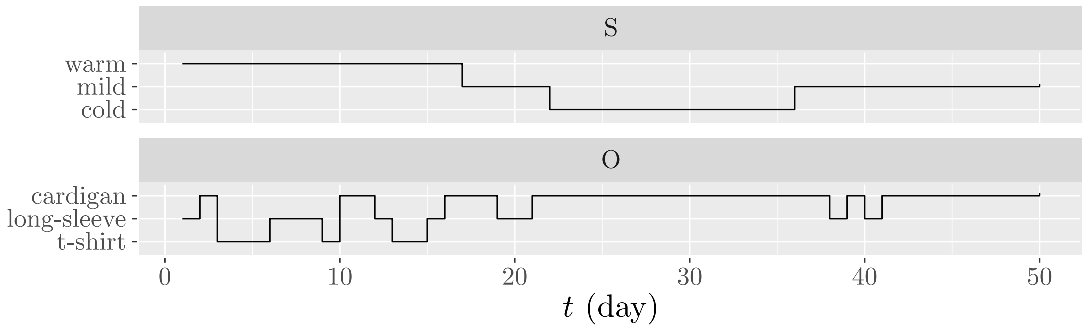
We illustrate with an example inspired by Ravindranath (2019). Alice and her friend Bob live far away from each other, but they have a video call every day for a period of fifty days. They decided to have a guessing game, where Bob should guess whether it is cold, mild or warm where Alice lives, based on which of her three sets of type of clothing she is wearing; a t-shirt, a long sleeve, or a cardigan. Bob knows that the temperature where Alice lives is very stable and have a low probability of changing, and he knows that Alice is often cold and generally prefers to wear a long-sleeve or a cardigan, but might occasionally wear a t-shirt on warm days. Bob has decided to use a HMM for this problem and have come up with the following description: Let $S^{(t)}$ be the temperature of where Alice lives where the sample space is $\{1, 2, 3\}$ according to cold, mild and warm weather, respectively, and let $O^{(t)}$ be the type of clothing Alice wears during their video meetings with the sample space $\{1, 2, 3\}$ according to t-shirt, long-sleeve and cardigan, respectively. The parameters of the HMM are

$$\boldsymbol{\pi} = \begin{bmatrix} 1/3 \\ 1/3 \\ 1/3 \end{bmatrix}, \quad \mathbf{P} = \begin{bmatrix} 0.97 & 0.02 & 0.01 \\ 0.01 & 0.98 & 0.01 \\ 0.01 & 0.02 & 0.97 \end{bmatrix} \quad \mathbf{E} = \begin{bmatrix} 0 & 0.05 & 0.95 \\ 0 & 0.2 & 0.8 \\ 0.2 & 0.5 & 0.3 \end{bmatrix}. \tag{3.10}$$

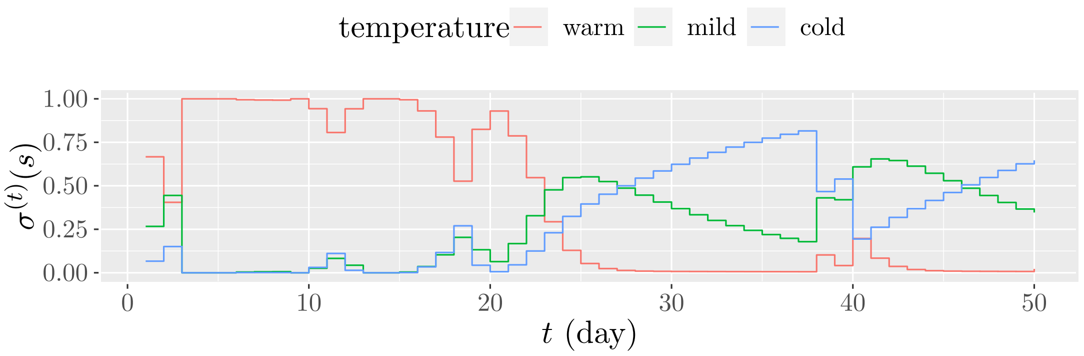
Remember that, the rows of the probability matrices correspond to the condition, and the column correspond to the outcome of the variable we compute the probability of, e.g. the probability of the temperature being mild and warm tomorrow given that the weather was mild today is 0.98 and 0.01, respectively. We also see that Alice only wears t-shirt on warm days, as there is a non-zero probability of her wearing a t-shirt only when it is warm.

Of course, Alice has kept track of the temperature every day, and also what type of clothing she has had, so she can compare the truth with Bob's estimate. This is shown in Figure 3.4a. Here we see that it is warm the first sixteen days, and Alice is wearing all types of clothing during this period. On day seventeen through day twenty-one, it is mild, and Alice going back and forth wearing a long-sleeve and a cardigan. Then when it is cold, she consistently wears a cardigan until it is mild again on day thirty-six, then two of the remaining days, she wears a long-sleeve.

Bob has carefully computed the filtered estimate of the temperature at each day using his a priori knowledge from $\boldsymbol{\pi}$, $\mathbf{P}$ and $\mathbf{E}$ and his observations on what type of clothing Alice has worn. This is shown in Figure 3.4b. Here Bob is very certain that it is warm the first days (except for the second day where Alice wears a cardigan), and is 100 percent certain the days Alice was wearing a t-shirt. It is not before day twenty-three Bob changes his opinion about the temperature where he believes it now is mild. However, we also see that his belief about the temperature being cold starts to increase on day twenty-one. It is not before day twenty-seven until Bob changes his belief about the temperature from being mild to cold. This is a very typical behavior in filtering, when something changes, we might still be a bit conservative and stick with our current belief as the new observation might be a one-time-occurrence. It is not until we see that the change is persistent, we change our belief.



(a) The true temperature on each day where Alice lives (S) and what type of clothing Alice is wearing that day (O).



(b) Filtered estimate on what the temperature is where Alice lives based on what type of clothing she was wearing during her and Bob's video meetings.

Figure 3.4: HMM example with Alice and Bob.

3.3 Continuous - KF

The continuous case looks a bit different, but the main idea remains the same. The filtering algorithm for LDSs that are CLGs has a special name, *Kalman filter* (KF) which is named after Kalman (1960) who was one of the primary developers of its theory.

CLGs are favorable to work with because of their nice properties of being both Gaussian and linear. They are easy to combine, i.e. computing the joint distribution from separate CLGs, they are easy separate, i.e. either marginalizing them, or computing the posterior distribution. We start by showing the most important properties of multivariate Gaussian distributions, followed by the most important properties of CLGs before we show the filtering approach for LDSs.

The PDF of the multivariate Gaussian distribution with mean $\boldsymbol{\mu}$ and variance $\boldsymbol{\Sigma}$ is most commonly defined in moment form as follows:

$$f(\mathbf{y}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) = |2\pi\boldsymbol{\Sigma}|^{-\frac{1}{2}} \exp\left\{-\frac{1}{2}(\mathbf{y}-\boldsymbol{\mu})^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{y}-\boldsymbol{\mu})\right\}, \quad \mathbf{y} \in \mathbb{R}^N, \tag{3.11}$$

where $|\mathbf{Q}|$ is the determinant of a square matrix $\mathbf{Q}$. Let $\mathbf{Y}$ be a random vector of length greater than one. Then we can partition it, its mean and covariance matrix at as follows:

$$\mathbf{Y} = \begin{bmatrix} \mathbf{Y}_A \\ \mathbf{Y}_B \end{bmatrix} \quad \boldsymbol{\mu} = \begin{bmatrix} \boldsymbol{\mu}_A \\ \boldsymbol{\mu}_B \end{bmatrix} \quad \boldsymbol{\Sigma} = \begin{bmatrix} \boldsymbol{\Sigma}_{AA} & \boldsymbol{\Sigma}_{AB} \\ \boldsymbol{\Sigma}_{BA} & \boldsymbol{\Sigma}_{BB} \end{bmatrix}, \tag{3.12}$$

where the length of $\mathbf{Y}_A$ matches the length of $\boldsymbol{\mu}_A$ and the number of rows and columns of $\boldsymbol{\Sigma}_{AA}$. The first nice property of multivariate Gaussian distributions is they are easy to marginalize. For instance, we have that $\mathbf{Y}_A \sim \mathcal{N}(\boldsymbol{\mu}_A, \boldsymbol{\Sigma}_{AA})$. So by only inspecting the relevant part of the full mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$ in Expression (3.12), we automatically find the parameters of the marginal vector $\mathbf{Y}_A$, with no computation involved, and similarly for $\mathbf{Y}_B$.

Not only is the marginal partition Gaussian, but one partition conditioned on the other partition is also Gaussian. That is, let the random vector $\mathbf{Y}$ be Gaussian $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ with length greater than one, and let it be partitioned according to Expression (3.12). Then $\mathbf{Y}_A \mid \{\mathbf{Y}_B = \mathbf{y}_B\}$ has distribution $\mathcal{N}(\boldsymbol{\mu}_{A|B}, \boldsymbol{\Sigma}_{A|B})$ where

$$\begin{aligned} \boldsymbol{\mu}_{A|B} &= \boldsymbol{\mu}_A + \boldsymbol{\Sigma}_{AB}\boldsymbol{\Sigma}_{BB}^{-1}(\mathbf{y}_B - \boldsymbol{\mu}_B), \\ \boldsymbol{\Sigma}_{A|B} &= \boldsymbol{\Sigma}_{AA} - \boldsymbol{\Sigma}_{AB}\boldsymbol{\Sigma}_{BB}^{-1}\boldsymbol{\Sigma}_{BA}. \end{aligned} \tag{3.13}$$

Now, consider a simple CLG BN consisting of the variables $\mathcal{X} = \{\mathbf{Y}_A, \mathbf{Y}_B\}$, where $\mathbf{Y}_A$ is the parent of $\mathbf{Y}_B$. That is, let

$$\begin{aligned} \mathbf{Y}_B \mid \{\mathbf{Y}_A = \mathbf{y}_A\} &\sim \mathcal{N}(\mathbf{C}\mathbf{y}_A + \mathbf{d}, \boldsymbol{\Sigma}_{B|A}), \\ \mathbf{Y}_A &\sim \mathcal{N}(\boldsymbol{\mu}_A, \boldsymbol{\Sigma}_{AA}), \end{aligned} \tag{3.14}$$

where $\mathbf{C}$ and $\mathbf{d}$ are fixed conformable matrix and vector, respectively. Then the joint distribution of $\mathbf{Y}_A$ and $\mathbf{Y}_B$ is given by

$$\mathbf{Y} = \begin{bmatrix} \mathbf{Y}_A \\ \mathbf{Y}_B \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \boldsymbol{\mu}_A \\ \mathbf{C}\boldsymbol{\mu}_A + \mathbf{d} \end{bmatrix}, \begin{bmatrix} \boldsymbol{\Sigma}_{AA} & \boldsymbol{\Sigma}_{AA}\mathbf{C}^\mathsf{T} \\ \mathbf{C}\boldsymbol{\Sigma}_{AA} & \boldsymbol{\Sigma}_{B|A} + \mathbf{C}\boldsymbol{\Sigma}_{AA}\mathbf{C}^\mathsf{T} \end{bmatrix} \right). \tag{3.15}$$

We see that if we wish to marginalize $\mathbf{Y}_A$ from $\mathbf{Y}$ in Expression (3.15), we get back the original distribution $\mathcal{N}(\boldsymbol{\mu}_A, \boldsymbol{\Sigma}_{AA})$. We also now see the marginal distribution of $\mathbf{Y}_B$:

$$\mathbf{Y}_B \sim \mathcal{N}(\mathbf{C}\boldsymbol{\mu}_A + \mathbf{d}, \boldsymbol{\Sigma}_{B|A} + \mathbf{C}\boldsymbol{\Sigma}_{AA}\mathbf{C}^\mathsf{T}), \tag{3.16}$$

which we can confirm by using the law of total expectation and variance:

$$\begin{aligned} \mathrm{E}[\mathbf{Y}_B] &= \mathrm{E}[\mathrm{E}(\mathbf{Y}_B \mid \mathbf{Y}_A)] = \mathrm{E}[\mathbf{C}\mathbf{Y}_A + \mathbf{d}] \\ &= \mathbf{C}\,\mathrm{E}[\mathbf{Y}_A] + \mathbf{d} = \mathbf{C}\boldsymbol{\mu}_A + \mathbf{d} \\ \mathrm{Var}[\mathbf{Y}_B] &= \mathrm{E}[\mathrm{Var}(\mathbf{Y}_B \mid \mathbf{Y}_A)] + \mathrm{Var}[\mathrm{E}(\mathbf{Y}_B \mid \mathbf{Y}_A)], = \mathrm{E}[\boldsymbol{\Sigma}_{B|A}] + \mathrm{Var}[\mathbf{C}\mathbf{Y}_A + \mathbf{d}] \\ &= \boldsymbol{\Sigma}_{B|A} + \mathbf{C}\,\mathrm{Var}[\mathbf{Y}_A]\mathbf{C}^\mathsf{T} = \boldsymbol{\Sigma}_{B|A} + \mathbf{C}\boldsymbol{\Sigma}_{AA}\mathbf{C}^\mathsf{T}. \end{aligned}$$

We can also confirm the off-diagonal terms in the covariance matrix in Expression (3.15) as follows

$$\mathrm{Cov}[\mathbf{Y}_B, \mathbf{Y}_A] = \mathrm{Cov}[\mathrm{E}(\mathbf{Y}_B \mid \mathbf{Y}_A), \mathbf{Y}_A] = \mathrm{Cov}[\mathbf{C}\mathbf{Y}_A + \mathbf{d}, \mathbf{Y}_A] = \mathbf{C}\,\mathrm{Var}[\mathbf{Y}_A] = \mathbf{C}\boldsymbol{\Sigma}_{AA},$$

which coincides of the lower left partition of the covariance matrix in Expression (3.15).

Finally, we are interested in the conditional distribution $\mathbf{Y}_A \mid \mathbf{Y}_B$, which is the opposite of what we are given in Expression (3.14). Here, we can think of $\mathbf{Y}_A$ being the hidden state $\mathbf{X}$, and $\mathbf{Y}_B$ is the observation $\mathbf{O}$ which depends on the hidden state. We find the conditional distribution $\mathbf{Y}_A \mid \{\mathbf{Y}_B = \mathbf{y}_B\} \sim \mathcal{N}(\boldsymbol{\mu}_{A|B}, \boldsymbol{\Sigma}_{A|B})$ by combining the result in Expression (3.13) and (3.15):

$$\begin{aligned}
\boldsymbol{\mu}_{A|B} &= \boldsymbol{\mu}_A + \boldsymbol{\Sigma}_{AB}\boldsymbol{\Sigma}_{BB}^{-1}(\mathbf{y}_B - \boldsymbol{\mu}_B) \\
&= \boldsymbol{\mu}_A + (\boldsymbol{\Sigma}_{AA}\mathbf{C}^\mathsf{T})(\boldsymbol{\Sigma}_{B|A} + \mathbf{C}\boldsymbol{\Sigma}_{AA}\mathbf{C}^\mathsf{T})^{-1}(\mathbf{y}_B - (\mathbf{C}\boldsymbol{\mu}_A + \mathbf{d})) \qquad (3.17)\\
&= \boldsymbol{\mu}_A + \mathbf{K}(\mathbf{y}_B - (\mathbf{C}\boldsymbol{\mu}_A + \mathbf{d})) \\
\boldsymbol{\Sigma}_{A|B} &= \boldsymbol{\Sigma}_{AA} - \boldsymbol{\Sigma}_{AB}\boldsymbol{\Sigma}_{BB}^{-1}\boldsymbol{\Sigma}_{BA} \\
&= \boldsymbol{\Sigma}_{AA} - (\boldsymbol{\Sigma}_{AA}\mathbf{C}^\mathsf{T})(\boldsymbol{\Sigma}_{B|A} + \mathbf{C}\boldsymbol{\Sigma}_{AA}\mathbf{C}^\mathsf{T})^{-1}(\mathbf{C}\boldsymbol{\Sigma}_{AA}) \qquad (3.18)\\
&= (\mathbf{I} - \mathbf{K}\mathbf{C})\boldsymbol{\Sigma}_{AA} \\
\mathbf{K} &= \boldsymbol{\Sigma}_{AA}\mathbf{C}^\mathsf{T}(\boldsymbol{\Sigma}_{B|A} + \mathbf{C}\boldsymbol{\Sigma}_{AA}\mathbf{C}^\mathsf{T})^{-1}, \qquad (3.19)
\end{aligned}$$

where $\mathbf{I}$ is the identity matrix.

We now show the KF approach for the LDS. That is, we assume a state-observation model where the conditional distributions are all linear Gaussians. The dynamical state is the N-vector $\mathbf{Y}^{(t)}$ and the observation state is the L-vector $\mathbf{O}^{(t)}$. The model is fully described in Expression (2.10) and has the model parameters $\boldsymbol{\nu}$, $\boldsymbol{\Gamma}$, $\mathbf{A}$, $\mathbf{b}$, $\mathbf{Q}$, $\mathbf{C}$, $\mathbf{d}$ and $\mathbf{R}$.

Now, since $\mathbf{Y}$ is continuous, the filtered estimate is given by the density

$$\sigma^{(t)}(\mathbf{y}) = p_{\mathbf{Y}^{(t)}}(\mathbf{y} \mid \mathbf{o}^{(1:t)}) = f(\mathbf{y}; \boldsymbol{\mu}^{(t)}, \boldsymbol{\Sigma}^{(t)}), \quad \mathbf{y} \in \mathbb{R}^N.$$

The prediction step $\sigma^{(\cdot t+1)}(\cdot)$ is computed as

$$\begin{aligned}
\sigma^{(\cdot t+1)}(\mathbf{y}) &= \int_{\mathbf{z}\in\mathbb{R}^N} p_{\mathbf{Y}'}(\mathbf{y} \mid \mathbf{Y} = \mathbf{z})\sigma^{(t)}(\mathbf{z})\,\mathrm{d}\mathbf{z} \\
&= \int_{\mathbf{z}\in\mathbb{R}^N} p_{\mathbf{Y}^{(t+1)}}(\mathbf{y} \mid \mathbf{Y}^{(t)} = \mathbf{z}, \mathbf{o}^{(1:t)})p_{\mathbf{Y}^{(t)}}(\mathbf{z} \mid \mathbf{o}^{(1:t)})\,\mathrm{d}\mathbf{z} \\
&= \int_{\mathbf{z}\in\mathbb{R}^N} f(\mathbf{y}; \mathbf{A}\mathbf{z} + \mathbf{b}, \mathbf{Q})f(\mathbf{z}; \boldsymbol{\mu}^{(t)}, \boldsymbol{\Sigma}^{(t)})\,\mathrm{d}\mathbf{z} \\
&= f(\mathbf{y}; \boldsymbol{\mu}^{(\cdot t+1)}, \boldsymbol{\Sigma}^{(\cdot t+1)}), \quad \mathbf{y} \in \mathbb{R}^N,
\end{aligned}$$

where

$$\begin{aligned}
\boldsymbol{\mu}^{(\cdot t+1)} &= \mathbf{A}\boldsymbol{\mu}^{(t)} + \mathbf{b}; \\
\boldsymbol{\Sigma}^{(\cdot t+1)} &= \mathbf{Q} + \mathbf{A}\boldsymbol{\Sigma}^{(t)}\mathbf{A}^\mathsf{T}.
\end{aligned} \qquad (3.20)$$

Here, we used the result from Expression (3.14), (3.15) and (3.16), together with the conditional independence shown in Figure 3.3a.

The condition step $\sigma^{(t+1)}(\cdot)$ is computed as

$$\begin{aligned}\sigma^{(t+1)}(\mathbf{y}) &= p_{\mathbf{Y}^{(t+1)}}(\mathbf{y}, \mid \mathbf{o}^{(t+1)}, \mathbf{o}^{(1:t)}) \\ &\propto p(\mathbf{o}^{(t+1)} \mid \mathbf{Y}^{(t+1)} = \mathbf{y}, \mathbf{o}^{(1:t)}) p_{\mathbf{Y}^{(t+1)}}(\mathbf{y} \mid \mathbf{o}^{(1:t)}) \\ &= f(\mathbf{o}^{(t+1)}; \mathbf{C}\mathbf{y} + \mathbf{d}, \mathbf{R}) f(\mathbf{y}; \boldsymbol{\mu}^{(\cdot t+1)}, \boldsymbol{\Sigma}^{(\cdot t+1)}) \\ \Rightarrow \sigma^{(t+1)}(\mathbf{y}) &= f(\mathbf{y}; \boldsymbol{\mu}^{(t+1)}, \boldsymbol{\Sigma}^{(t+1)}), \quad \mathbf{y} \in \mathbb{R}^N,\end{aligned}$$

where

$$\begin{aligned}\mathbf{K}^{(t+1)} &= \boldsymbol{\Sigma}^{(\cdot t+1)} \mathbf{C}^{\mathsf{T}} (\mathbf{R} + \mathbf{C} \boldsymbol{\Sigma}^{(\cdot t+1)} \mathbf{C}^{\mathsf{T}})^{-1}; \\ \boldsymbol{\mu}^{(t+1)} &= \boldsymbol{\mu}^{(\cdot t+1)} + \mathbf{K}^{(t+1)} (\mathbf{o}^{(t+1)} - (\mathbf{C} \boldsymbol{\mu}^{(\cdot t+1)} + \mathbf{d})); \\ \boldsymbol{\Sigma}^{(t+1)} &= (\mathbf{I} - \mathbf{K}^{(t+1)} \mathbf{C}) \boldsymbol{\Sigma}^{(\cdot t+1)}.\end{aligned} \tag{3.21}$$

Here, we used the result from Expression (3.17), (3.18) and (3.19), together with the conditional independence shown in Figure 3.3b. Notice the matrix $\mathbf{K}^{(t+1)}$ in Expression (3.21). This is called the *Kalman gain*, and we can think of it as a weight that determines how important the new observation $\mathbf{o}^{(t+1)}$ is when estimating the mean $\boldsymbol{\mu}^{(t+1)}$, e.g. if the Kalman gain is a zero-matrix, then we do not include the observation in the calculation, we simply get $\boldsymbol{\mu}^{(t+1)} = \boldsymbol{\mu}^{(\cdot t+1)}$.

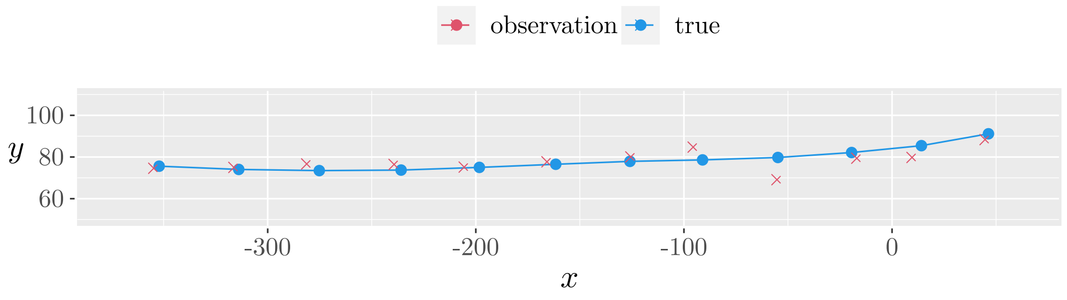
We illustrate the KF with an example, where we wish to track an object moving in the xy-plane. We let $\mathbf{Y}^{(t)} = (X^{(t)}, Y^{(t)}, V_x^{(t)}, V_y^{(t)})^{\mathsf{T}}$, where $X^{(t)}$, and $Y^{(t)}$ are (x, y)-coordinates, and $V_x^{(t)}$ and $V_y^{(t)}$ are the x and y-component velocities such that the absolute velocity is $V^{(t)} = \sqrt{(V_x^{(t)})^2 + (V_y^{(t)})^2}$. Also, we let $\mathbf{O}^{(t)} = (O_x^{(t)}, O_y^{(t)})^{\mathsf{T}}$, where $O_x^{(t)}$ and $O_y^{(t)}$ are the observed (x, y)-coordinates originating from GPS data. The parameters are

$$\begin{aligned}&\boldsymbol{\nu} = \begin{bmatrix}0\\0\\0\\0\end{bmatrix}, \quad \boldsymbol{\Gamma} = \begin{bmatrix}50^2 & 0 & 0 & 0\\ 0 & 50^2 & 0 & 0\\ 0 & 0 & 20^2 & 0\\ 0 & 0 & 0 & 20^2\end{bmatrix}; \\ &\mathbf{A} = \begin{bmatrix}1 & 0 & \Delta t & 0\\ 0 & 1 & 0 & \Delta t\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix}0\\0\\0\\0\end{bmatrix}, \quad \mathbf{Q} = \begin{bmatrix}0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\end{bmatrix}; \\ &\mathbf{C} = \begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\end{bmatrix}, \quad \mathbf{d} = \begin{bmatrix}0\\0\end{bmatrix}, \quad \mathbf{R} = \begin{bmatrix}5^2 & 0\\ 0 & 5^2\end{bmatrix},\end{aligned} \tag{3.22}$$

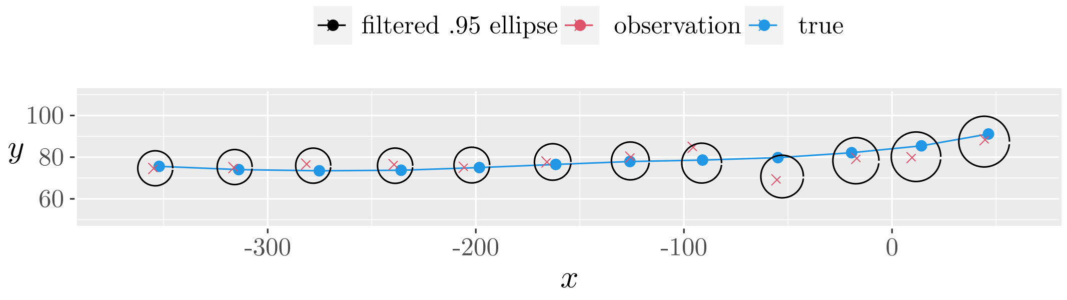
where Δt is the time difference between each time step, which we set equal to 1. The mean of $\mathbf{Y}^{(t+1)} \mid \{\mathbf{Y}^{(t)} = \mathbf{y}\}$ is thus

$$\mathbf{A}\mathbf{y} + \mathbf{b} = \begin{bmatrix}x + \Delta t \cdot v_x\\ y + \Delta t \cdot v_y\\ v_x\\ v_y\end{bmatrix},$$

which satisfies the assumption of a constant velocity model. But this is only an approximation, as the covariance matrix is non-zero for v_x and v_y. This is to allow for some Gaussian noise to the velocity-components account for the approximation we make. A simulation of this model is shown in Figure 3.5a, where we see the moving object starting from the right side, and moves towards the left side. The motion is quite straight as expected because of the low variance of the velocity components in $\mathbf{Q}$. We see that the



(a) The true position in blue, and observed position in red.



(b) Filtered estimate added, shown as 0.95 confidence ellipses.

Figure 3.5: KF example, where we are tracking the position of an object moving in the xy-plane.

observations (red crosses) jumps on each side of the trajectory due to the noise in $\mathbf{R}$.

The estimate from the KF is shown in Figure 3.5b where we have used 0.95 confidence ellipses. The area of the ellipses are larger on the right side than the left side. This is because we do not have any knowledge of the direction of the moving object yet. As we start to get more observations, we are more certain in which direction the object is moving, and the area of the ellipse is reduced.

3.4 Hybrid - SLDS

Getting the filtered estimates on HMMs and KFs are straight forward. For the discrete case with HMMs we get a complexity of $\mathcal{O}(tM^2)$, where t is the number of time step and M is the number of hidden states. For the continuous case with KFs we get a complexity of $\mathcal{O}(t \max\{N, L\}^3)$, where N is the length of the hidden vector $\mathbf{Y}$ and L is the length of the observed vector $\mathbf{O}$. For hybrid networks, where discrete variables may only have discrete parents while continuous variables can have both continuous and discrete parents, computing the closed form filtered estimate turns out to be problematic, and one must instead turn to heuristic approaches.

3.4.1 The problem with hybrid DBNs

To demonstrate the problem, we use the DBN shown in Figure 3.6. The conditional

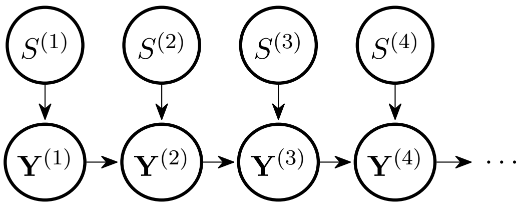


Figure 3.6: Simple hybrid DBN, where S is discrete and $\mathbf{Y}$ is continuous.

probability distributions for this network is given by

$$\begin{aligned} \mathbf{Y}^{(1)} \mid \{S^{(1)} = s\} &\sim \mathcal{N}(\boldsymbol{\nu}, \boldsymbol{\Gamma}_s) && \text{(Initial distribution)} \\ \mathbf{Y}' \mid \{S' = s, \mathbf{Y} = \mathbf{y}\} &\sim \mathcal{N}(\mathbf{A}_s\mathbf{y} + \mathbf{b}_s, \mathbf{Q}_s) && \text{(Transition model)} \\ P(S' \mid S) = P(S') &= (p_i) \in \mathbb{R}^M && \text{(Transition/Marginal model).} \end{aligned} \tag{3.23}$$

In this network there is no observation model, so the forward algorithm reduces to only performing the prediction steps recursively, and skips the condition step. We have also omitted the edges between the discrete variables to make the following computation easier to follow.

Say we wish to compute the marginal distribution of $\mathbf{Y}^{(t)}$, which we start off by computing the base case:

$$\sigma^{(1)}(s, \mathbf{y}) = P(S^{(1)} = s) \times p_{\mathbf{Y}^{(1)}}(\mathbf{y} \mid S^{(1)} = s) = p_s f(\mathbf{y}; \boldsymbol{\nu}_s, \boldsymbol{\Gamma}_s).$$

If we wish to find the marginal distribution of $\mathbf{Y}^{(1)}$, we can simply sum over s, that is,

$$\sigma^{(1)}(\mathbf{y}) = \sum_{s=1}^{M} \sigma^{(1)}(s, \mathbf{y}) = \sum_{s=1}^{M} p_s f(\mathbf{y}; \boldsymbol{\nu}_s, \boldsymbol{\Gamma}_s),$$

which we can see is a Gaussian mixture, consisting of a weighted combination of M multivariate Gaussian densities. Let us continue, where we now try to compute the filtered estimate for $t = 2$.

Now, following the algorithm, we compute the prediction step as shown in Expression (3.4), to get the predicted, (in this case, also filtered) estimate at $t = 2$. Note, that the full transition model $P(S') = P(S' \mid S)$ and $p_{\mathbf{Y}'}(\mathbf{Y}' \mid S', \mathbf{Y})$, can be written compactly as follows:

$$\mathcal{P}(S')\mathcal{P}(\mathbf{Y}' \mid S', \mathbf{Y}) = \mathcal{P}(S' \mid S, \mathbf{Y})\mathcal{P}(Y' \mid S', S, \mathbf{Y}) = \mathcal{P}(S', \mathbf{Y}' \mid S, Y),$$

where we use calligraphic $\mathcal{P}(Y)$ to cover both probabilities $P(Y = y)$ and densities $p_Y(y)$. The filtered estimate is:

$$\begin{aligned}
\sigma^{(2)}(s, \mathbf{y}) &= \sum_{j=1}^{M} \int_{\mathbb{R}^N} \mathcal{P}(S' = s, \mathbf{Y}' = \mathbf{y} \mid S = j, \mathbf{Y} = \mathbf{z})\sigma^{(1)}(j, \mathbf{z})\,\mathrm{d}\mathbf{z} \\
&= \sum_{j=1}^{M} \int_{\mathbb{R}^N} p_s f(\mathbf{y}; \mathbf{A}_s\mathbf{z} + \mathbf{b}_s, \mathbf{Q}_s) p_j f(\mathbf{z}; \boldsymbol{\nu}_j, \boldsymbol{\Gamma}_j)\,\mathrm{d}\mathbf{z} \\
&= p_s \sum_{j=1}^{M} p_j \int_{\mathbb{R}^N} f(\mathbf{y}; \mathbf{A}_s\mathbf{z} + \mathbf{b}_s, \mathbf{Q}_s) f(\mathbf{z}; \boldsymbol{\nu}_j, \boldsymbol{\Gamma}_j)\,\mathrm{d}\mathbf{z} \\
&= p_s \sum_{j=1}^{M} p_j f(\mathbf{y}; \mathbf{A}_s\boldsymbol{\nu}_j + \mathbf{b}_s, \mathbf{Q}_s + \mathbf{A}\boldsymbol{\Gamma}_j\mathbf{A}^{\mathsf{T}}),
\end{aligned}$$

where we in the last equality used the result from Expression (3.16). Now, we can see that the marginal distribution of $\mathbf{Y}^{(2)}$ is an even larger mix:

$$\sigma^{(2)}(y) = \sum_{s=1}^{M} \sigma^{(2)}(s, \mathbf{y}) = \sum_{s=1}^{M}\sum_{j=1}^{M} p_s p_j f(\mathbf{y}; \mathbf{A}_s\boldsymbol{\nu}_j + \mathbf{b}_s, \mathbf{Q}_s + \mathbf{A}_s\boldsymbol{\Gamma}_j\mathbf{A}^{\mathsf{T}}),$$

which has the form

$$\sigma^{(2)}(\mathbf{y}) = \sum_{k=1}^{M^2} w_k^{(2)} f(\mathbf{y}; \boldsymbol{\mu}_k^{(2)}, \boldsymbol{\Sigma}_k^{(2)}),$$

a combination of M^2 Gaussian densities! For the general case, we have that the filtered estimate for Y at time t has the following form

$$\sigma^{(t)}(\mathbf{y}) = \sum_{k=1}^{M^t} w_k^{(t)} f(\mathbf{y}; \boldsymbol{\mu}_k^{(t)}, \boldsymbol{\Sigma}_k^{(t)}),$$

a mixture of M^t Gaussian densities! The mixture size grows exponentially with time. For only $M = 2$ classes, the filtered estimate for $\mathbf{Y}^{(30)}$ becomes a mixture of more than *one billion* Gaussian densities! Thus, exact inference on DBNs where continuous variables have discrete parents is not feasible. There are many proposed solutions for this, including both deterministic and stochastic approaches (Blom & Bar-Shalom, 1988; Murphy, 1998; Lerner, 2003; Koller & Friedman, 2009). The main idea for the deterministic approach is to find a way to set an upper bound to the number of Gaussian mixtures per time step. This is explained in detail in Section 3.4.3. First, we find the analytical expressions for exact inference of the SLDS.

3.4.2 Exact inference

We remind ourselves that the variables in the SLDS has the following CPDs:

$$\begin{aligned}
P(S^{(1)}) &= \boldsymbol{\pi} = (\pi_i) \in \mathbb{R}^M && \text{(Initial distribution)} \\
\mathbf{Y}^{(1)} \mid \{S^{(1)} = s\} &\sim \mathcal{N}(\boldsymbol{\nu}_s, \boldsymbol{\Gamma}_s) && \text{(Initial distribution)} \\
P(S' \mid S) &= \mathbf{P} = (p_{ij}) \in \mathbb{R}^{M\times M} && \text{(Transition model)} \\
\mathbf{Y}' \mid \{S' = s', \mathbf{Y} = \mathbf{y}\} &\sim \mathcal{N}(\mathbf{A}_{s'}\mathbf{y} + \mathbf{b}_{s'}, \mathbf{Q}_{s'}) && \text{(Transition model)} \\
\mathbf{O} \mid \{S = s, \mathbf{Y} = \mathbf{y}\} &\sim \mathcal{N}(\mathbf{C}_s\mathbf{y} + \mathbf{d}_s, \mathbf{R}_s) && \text{(Observation model).}
\end{aligned} \tag{3.24}$$

The general form of the filtered estimate is

$$\sigma^{(t)}(s) = P(S^{(t)} = s \mid \mathbf{o}^{(1:t)}) = p_s^{(t)}, \quad s = 1, \ldots, M \tag{3.25}$$

$$\sigma^{(t)}(\mathbf{y}) = p_{\mathbf{Y}^{(t)}}(\mathbf{y} \mid \mathbf{o}^{(1:t)}) = \sum_{k=1}^{M^t} w_k^{(t)} f(\mathbf{y}; \boldsymbol{\mu}_k^{(t)}, \boldsymbol{\Sigma}_k^{(t)}), \quad \mathbf{y} \in \mathbb{R}^N \tag{3.26}$$

where the index k uniquely represents each possible history

$$s^{(1:t)}(k) = \{S^{(t)} = s^{(t)}(k), S^{(t-1)} = s^{(t-1)}(k), \ldots, S^{(1)} = s^{(1)}(k)\}. \tag{3.27}$$

The above representation can be made possible with the following bijective relation between the index k and the history,

$$s^{(1:t)}(k) = \{s^{(i)}(k) : i = 1, \ldots, t\} \quad \text{where} \quad s^{(t)}(k) = \left(\left\lfloor \frac{k-1}{M^{t-1}} \right\rfloor \mod M \right) + 1. \tag{3.28}$$

We also make use of a different parameterization of the indices when applying the forward algorithm. That is, $j \mid \{S^{(t)} = s\}$, where j represents the history at times $1, \ldots, t-1$, and $S^{(t)} = s$ represent the history at time t. There is a bijection between the indices $j \mid \{S^{(t)} = s\}$ and k as follows:

$$\begin{aligned}
&k \mapsto j \mid \{S^{(t)} = s\} \quad \text{for} \quad \begin{cases} s &= \lfloor \frac{k-1}{M^{t-1}} \rfloor + 1, \\ j &= \left((k-1) \mod M^{t-1}\right) + 1. \end{cases} \\
&j \mid \{S^{(t)} = s\} \mapsto k \quad \text{for} \quad k = (s-1)M^{t-1} + j
\end{aligned} \tag{3.29}$$

Table 3.1 demonstrates the M^t possible histories for $t = 3$ and $M = 2$ along with the indices $1 \leq k \leq M^t$, $1 \leq j \leq M^{t-1}$ and $1 \leq s \leq M$ for each respective history. To not get overwhelmed by long subscripts, we often reduce $j \mid \{S^{(t)} = s\}$ to simply $j \mid s$, since the time step is implicitly given by the variables we are subscripting, e.g.

$$v^{(t)}_{j \mid \{S^{(t)} = s\}} = v^{(t)}_{j \mid s}.$$

Table 3.1: The index k represents the k'th possible history of the states $S^{(t)}$ for $t = 1, 2, 3$, where we have $M = 2$ possible states per time step.

k	$j \mid \{S^{(3)} = s\}$	$s^{(3)}(k)$	$s^{(2)}(k)$	$s^{(1)}(k)$	$s^{(1:3)}(k)$
1	$1 \mid \{S^{(3)} = 1\}$	1	1	1	$\{S^{(3)} = 1, S^{(2)} = 1, S^{(1)} = 1\}$
2	$2 \mid \{S^{(3)} = 1\}$	1	1	2	$\{S^{(3)} = 1, S^{(2)} = 1, S^{(1)} = 2\}$
3	$3 \mid \{S^{(3)} = 1\}$	1	2	1	$\{S^{(3)} = 1, S^{(2)} = 2, S^{(1)} = 1\}$
4	$4 \mid \{S^{(3)} = 1\}$	1	2	2	$\{S^{(3)} = 1, S^{(2)} = 2, S^{(1)} = 2\}$
5	$1 \mid \{S^{(3)} = 2\}$	2	1	1	$\{S^{(3)} = 2, S^{(2)} = 1, S^{(1)} = 1\}$
6	$2 \mid \{S^{(3)} = 2\}$	2	1	2	$\{S^{(3)} = 2, S^{(2)} = 1, S^{(1)} = 2\}$
7	$3 \mid \{S^{(3)} = 2\}$	2	2	1	$\{S^{(3)} = 2, S^{(2)} = 2, S^{(1)} = 1\}$
8	$4 \mid \{S^{(3)} = 2\}$	2	2	2	$\{S^{(3)} = 2, S^{(2)} = 2, S^{(1)} = 2\}$

When we apply the forward algorithm, we combine the filtered estimate for $S^{(t)}$ and $\mathbf{Y}^{(t)}$ as follows

$$\sigma^{(t)}(s, \mathbf{y}) = \sigma^{(t)}(s)\sigma^{(t)}(\mathbf{y} \mid s) = p_s^{(t)} \sum_{j=1}^{M^{t-1}} v_{j|s}^{(t)} f\left(\mathbf{y}; \boldsymbol{\mu}_{j|s}^{(t)}, \boldsymbol{\Sigma}_{j|s}^{(t)}\right), \tag{3.30}$$

Here, the Gaussian mixture is conditioned on the most recent possible state $S^{(t)} = s$. This is why it is useful to have an alternative representation of the index k. We show the relation between the weights conditioned on $S^{(t)}$,

$$v_{j|s}^{(t)} = P(s^{(1:(t-1))}(j) \mid S^{(t)} = s, \mathbf{o}^{(1:t)}), \tag{3.31}$$

and the unconditioned weights

$$w_k^{(t)} = P(s^{(1:t)}(k) \mid \mathbf{o}^{(1:t)}), \tag{3.32}$$

by deriving the marginal filtered estimate $\sigma^{(t)}(\mathbf{y})$ from the joint $\sigma^{(t)}(s, \mathbf{y})$:

$$\begin{aligned}
\sigma^{(t)}(\mathbf{y}) &= \sum_{s=1}^{M} \sigma^{(t)}(s)\sigma^{(t)}(\mathbf{y} \mid s) \\
&= \sum_{s=1}^{M} p_s^{(t)} \sum_{j=1}^{M^{t-1}} v_{j|s}^{(t)} f\left(\mathbf{y}; \boldsymbol{\mu}_{j|s}^{(t)}, \boldsymbol{\Sigma}_{j|s}^{(t)}\right) \\
&= \sum_{s=1}^{M} \sum_{j=1}^{M^{t-1}} \sigma^{(t)}(s) v_{j|s}^{(t)} f\left(\mathbf{y}; \boldsymbol{\mu}_{j|s}^{(t)}, \boldsymbol{\Sigma}_{j|s}^{(t)}\right) \\
&= \sum_{k=1}^{M^t} \sigma^{(t)}(s^{(t)}(k)) v_k^{(t)} f(\mathbf{y}; \boldsymbol{\mu}_k^{(t)}, \boldsymbol{\Sigma}_k^{(t)}) \\
&= \sum_{k=1}^{M^t} w_k^{(t)} f(\mathbf{y}; \boldsymbol{\mu}_k^{(t)}, \boldsymbol{\Sigma}_k^{(t)}),
\end{aligned}$$

that is,

$$w_k^{(t)} = \sigma^{(t)}(s^{(t)}(k))v_k^{(t)}. \tag{3.33}$$

As we saw in Section 3.4.1, the increase of Gaussian components happens in the prediction step, so the predicted estimate has the following form

$$\sigma^{(\cdot t+1)}(s,\mathbf{y}) = p_s^{(\cdot t+1)} \sum_{j=1}^{M^t} v_{j|s}^{(\cdot t+1)} f\left(\mathbf{y}; \boldsymbol{\mu}_{j|s}^{(\cdot t+1)}, \boldsymbol{\Sigma}_{j|s}^{(\cdot t+1)}\right), \tag{3.34}$$

$$\sigma^{(\cdot t+1)}(s) = p_s^{(\cdot t+1)}, \quad s = 1,\dots,M, \tag{3.35}$$

$$\sigma^{(\cdot t+1)}(\mathbf{y}) = \sum_{k=1}^{M^{t+1}} w_k^{(\cdot t+1)} f(\mathbf{y}; \boldsymbol{\mu}_k^{(\cdot t+1)}, \boldsymbol{\Sigma}_k^{(\cdot t+1)}), \quad \mathbf{y} \in \mathbb{R}^N. \tag{3.36}$$

That is, the predicted estimate of $Y^{(t+1)}$ is a mixture of M^{t+1} Gaussian densities. Now we have the form of the filtered estimate, and the predicted estimate given in Expression (3.30) and (3.34). From the general recursive formula of the predicted step given by Expression (3.4), we have that

$$\begin{aligned}
\sigma^{(\cdot t+1)}(s,\mathbf{y}) &= \sum_{m=1}^{M} \int_{\mathbb{R}^N} \mathcal{P}(S' = s, \mathbf{Y}' = \mathbf{y} \mid S = m, \mathbf{Y} = \mathbf{z})\sigma^{(t)}(m,\mathbf{z})\,\mathrm{d}\mathbf{z} \\
&= \sum_{m=1}^{M} \int_{\mathbb{R}^N} p_{ms} f(\mathbf{y}; \mathbf{A}_s\mathbf{z} + \mathbf{b}_s, \mathbf{Q}_s) p_m^{(t)} \sum_{n=1}^{M^{t-1}} v_{n|m}^{(t)} f\left(\mathbf{z}; \boldsymbol{\mu}_{n|m}^{(t)}, \boldsymbol{\Sigma}_{n|m}^{(t)}\right) \mathrm{d}\mathbf{z} \\
&= \sum_{m=1}^{M} p_{ms} p_m^{(t)} \sum_{n=1}^{M^{t-1}} v_{n|m}^{(t)} \int_{\mathbb{R}^N} f(\mathbf{y}; \mathbf{A}_s\mathbf{z} + \mathbf{b}_s, \mathbf{Q}_s) f\left(\mathbf{z}; \boldsymbol{\mu}_{n|m}^{(t)}, \boldsymbol{\Sigma}_{n|m}^{(t)}\right) \mathrm{d}\mathbf{z} \\
&= \sum_{m=1}^{M} \sum_{n=1}^{M^{t-1}} p_{ms} p_m^{(t)} v_{n|m}^{(t)} f\left(\mathbf{y}; \mathbf{A}_s\boldsymbol{\mu}_{n|m}^{(t)} + \mathbf{b}_s, \mathbf{Q}_s + \mathbf{A}_s\boldsymbol{\Sigma}_{n|m}^{(t)}\mathbf{A}_s^{\mathsf{T}}\right) \\
&= \sum_{j=1}^{M^t} \Bigg[P(S' = s \mid S = s^{(t)}(j))\sigma^{(t)}(s^{(t)}(j)) \\
&\qquad\qquad v_j^{(t)} f(\mathbf{y}; \mathbf{A}_s\boldsymbol{\mu}_j^{(t)} + \mathbf{b}_s, \mathbf{Q}_s + \mathbf{A}_s\boldsymbol{\Sigma}_j^{(t)}\mathbf{A}_s^{\mathsf{T}}) \Bigg],
\end{aligned}$$

where we in the last equality applied the index mapping given by Expression (3.29). We are close to having the predicted estimate on the form given in Expression (3.34). We have

$$\boldsymbol{\mu}_{j|s}^{(\cdot t+1)} = \mathbf{A}_s\boldsymbol{\mu}_j^{(t)} + \mathbf{b}_s, \tag{3.37}$$

$$\boldsymbol{\Sigma}_{j|s}^{(\cdot t+1)} = \mathbf{Q}_s + \mathbf{A}_s\boldsymbol{\Sigma}_j^{(t)}\mathbf{A}_s^{\mathsf{T}}, \tag{3.38}$$

for $j = 1, \ldots, M^t$ and $s = 1, \ldots, M$, but we still need the predicted estimate $p_s^{(\cdot t+1)}$ and the weights $v_{j|s}^{(\cdot t+1)}$. We start by finding $p_s^{(\cdot t+1)}$ by integrating out $Y^{(t+1)}$ from $\sigma^{(\cdot t+1)}(s, \mathbf{y})$.

$$\begin{aligned}
p_s^{(\cdot t+1)} &= \int_{\mathbb{R}^N} \sigma^{(\cdot t+1)}(s, \mathbf{y}) \, \mathrm{d}\mathbf{y} \\
&= \sum_{m=1}^{M} p_{ms} p_m^{(t)} \int_{\mathbb{R}^N} \sum_{n=1}^{M^{t-1}} v_{n|m}^{(t)} f\left(\mathbf{y}; \mathbf{A}_s \boldsymbol{\mu}_{n|m}^{(t)} + \mathbf{b}_s, \mathbf{Q}_s + \mathbf{A}_s \boldsymbol{\Sigma}_{n|m}^{(t)} \mathbf{A}_s^\mathsf{T}\right) \mathrm{d}\mathbf{y} \\
&= \sum_{m=1}^{M} p_{ms} p_m^{(t)}
\end{aligned}$$

If we take a pause and look at this result, we see that this is simply

$$\sigma^{(\cdot t+1)}(s) = \sum_{m=1}^{M} P(S' = s \mid S = m) \sigma^{(t)}(S^{(t)} = m), \quad s = 1, \ldots, M,$$

which has exactly the same form as the original formula in Expression (3.4). The weights $v_{j|s}^{(\cdot t+1)}$ are obtained by conditioning on $S^{(t)}$:

$$\begin{aligned}
\sigma^{(\cdot t+1)}(\mathbf{y} \mid s) &= \frac{\sigma^{(\cdot t+1)}(s, \mathbf{y})}{\sigma^{(\cdot t+1)}(s)} \\
&= \frac{\sum_{j=1}^{M^t} P(S' = s \mid S = s^{(t)}(j)) \sigma^{(t)}(s^{(t)}(j)) v_j^{(t)} f(\mathbf{y}; \boldsymbol{\mu}_{j|s}^{(\cdot t+1)}, \boldsymbol{\Sigma}_{j|s}^{(\cdot t+1)})}{p_s^{(\cdot t+1)}} \\
&= \sum_{j=1}^{M^t} \left(\frac{P(S' = s \mid S = s^{(t)}(j)) \sigma^{(t)}(s^{(t)}(j))}{p_s^{(\cdot t+1)}} v_j^{(t)} \right) f(\mathbf{y}; \boldsymbol{\mu}_{j|s}^{(\cdot t+1)}, \boldsymbol{\Sigma}_{j|s}^{(\cdot t+1)}),
\end{aligned}$$

which gives us the weights,

$$v_{j|s}^{(\cdot t+1)} = \frac{P(S' = s \mid S = s^{(t)}(j)) \sigma^{(t)}(s^{(t)}(j))}{p_s^{(\cdot t+1)}} v_j^{(t)}, \quad s = 1, \ldots, M, \quad j = 1, \ldots, M^t. \tag{3.39}$$

That is, for every combinations of the previous history $s^{(1:t)}(j)$ for $j = 1, \ldots, M^t$, we get a new weight $v_{j|s}^{(\cdot t+1)}$ from $v_j^{(t)}$, that is reweighted by the term

$$\begin{aligned}
&\frac{P(S' = s \mid S = s^{(t)}(j)) \sigma^{(t)}(s^{(t)}(j))}{p_s^{(\cdot t+1)}} \\
&= \frac{P(S^{(t+1)} = s \mid S^{(t)} = s^{(t)}(j), \mathbf{o}^{(1:t)}) P(S^{(t)} = s^{(t)}(j) \mid \mathbf{o}^{(1:t)})}{P(S^{(t+1)} = s \mid \mathbf{o}^{(1:t)})} \\
&= P(s^{(t)}(j) \mid S^{(t+1)} = s, \mathbf{o}^{(1:t)}),
\end{aligned}$$

where we in the first equality used that $(S^{(t+1)} \perp \mathbf{o}^{(1:t)} \mid S^{(t)})$, as $\{S^{(t)}\} = \mathrm{Pa}(S^{(t+1)})$ and $\mathbf{o}^{(1:t)} \in \mathrm{NonDesc}(S^{(t+1)})$. When we multiply by $v_j^{(t)}$, we get

$$\begin{aligned}
v_{j|s}^{(\cdot t+1)} &= P(s^{(t)}(j) \mid S^{(t+1)} = s, \mathbf{o}^{(1:t)}) v_j^{(t)} \\
&= P(s^{(t)}(j) \mid S^{(t+1)} = s, \mathbf{o}^{(1:t)}) P(s^{(1:(t-1))}(j) \mid s^{(t)}(j), \mathbf{o}^{(1:t)}) \\
&= P(s^{(t)}(j) \mid S^{(t+1)} = s, \mathbf{o}^{(1:t)}) P(s^{(1:(t-1))}(j) \mid s^{(t)}(j), S^{(t+1)} = s, \mathbf{o}^{(1:t)}) \\
&= P(s^{(1:t)}(j) \mid S^{(t+1)} = s, \mathbf{o}^{(1:t)}),
\end{aligned}$$

which is exactly what is expected, a likelihood of having the history $s^{(1:t)}(j)$ given that the next state is $S^{(t+1)} = s$. In the third equality we used that $S^{(t)}$ d-separates $S^{(t+1)}$ from $S^{(1:(t-1))}$.

We have managed to express all parameters in the predicted estimate $\sigma^{(\cdot t+1)}(\cdot)$ in terms of the parameters in the filtered estimate $\sigma^{(t)}(\cdot)$. Now we do the same for the filtered estimate in the next time step $\sigma^{(t+1)}(\cdot)$ in terms of the parameters in $\sigma^{(\cdot t+1)}(\cdot)$. We start by using the result from Expression (3.5).

$$\begin{aligned}
\sigma^{(t+1)}(s, \mathbf{y}) &\propto p_{\mathbf{O}}(\mathbf{o}^{(t+1)} \mid S = s, \mathbf{Y} = \mathbf{y}) \sigma^{(\cdot t+1)}(s, \mathbf{y}) \\
&= f(\mathbf{o}^{(t+1)}; \mathbf{C}_s \mathbf{y} + \mathbf{d}_s, \mathbf{R}_s) p_s^{(\cdot t+1)} \sum_{j=1}^{M^t} v_{j|s}^{(\cdot t+1)} f(\mathbf{y}; \boldsymbol{\mu}_{j|s}^{(\cdot t+1)}, \boldsymbol{\Sigma}_{j|s}^{(\cdot t+1)}) \\
&= p_s^{(\cdot t+1)} \sum_{j=1}^{M^t} \widetilde{v}_{j|s}^{(t+1)} f(\mathbf{y}; \boldsymbol{\mu}_{j|s}^{(t+1)}, \boldsymbol{\Sigma}_{j|s}^{(t+1)}),
\end{aligned}$$

where

$$\boldsymbol{\mu}_{j|s}^{(t+1)} = \boldsymbol{\mu}_{j|s}^{(\cdot t+1)} + \mathbf{K}_{j|s}^{(t+1)}(\mathbf{o}^{(t+1)} - (\mathbf{C}_s \boldsymbol{\mu}_{j|s}^{(\cdot t+1)} + \mathbf{d}_s)) \tag{3.40}$$

$$\boldsymbol{\Sigma}_{j|s}^{(t+1)} = (\mathbf{I} - \mathbf{K}_{j|s}^{(t+1)} \mathbf{C}_s) \boldsymbol{\Sigma}_{j|s}^{(\cdot t+1)} \tag{3.41}$$

$$\mathbf{K}_{j|s}^{(t+1)} = \boldsymbol{\Sigma}_{j|s}^{(\cdot t+1)} \mathbf{C}_s^{\mathsf{T}} (\mathbf{R}_s + \mathbf{C}_s \boldsymbol{\Sigma}_{j|s}^{(\cdot t+1)} \mathbf{C}_s^{\mathsf{T}})^{-1} \tag{3.42}$$

$$\widetilde{v}_{j|s}^{(t+1)} = v_{j|s}^{(\cdot t+1)} f(\mathbf{o}^{(t+1)}; \mathbf{C}_s \boldsymbol{\mu}_{j|s}^{(\cdot t+1)} + \mathbf{d}_s, \mathbf{R}_s + \mathbf{C}_s \boldsymbol{\Sigma}_{j|s}^{(\cdot t+1)} \mathbf{C}_s^{\mathsf{T}}), \tag{3.43}$$

where we used that

$$\begin{aligned}
&f(\mathbf{o}^{(t+1)}; \mathbf{C}_s \mathbf{y} + \mathbf{d}_s, \mathbf{R}_s) f(\mathbf{y}; \boldsymbol{\mu}_{j|s}^{(\cdot t+1)}, \boldsymbol{\Sigma}_{j|s}^{(\cdot t+1)}) \\
&= p(\mathbf{o}^{(t+1)} \mid \mathbf{Y}^{(t+1)} = \mathbf{y}, S^{(t+1)} = s) p_{\mathbf{Y}^{(t+1)}}(\mathbf{y} \mid s^{(1:t)}(j), S^{(t+1)} = s, \mathbf{o}^{(1:t)}) \\
&= p(\mathbf{o}^{(t+1)} \mid \mathbf{Y}^{(t+1)} = \mathbf{y}, s^{(1:t)}(j), S^{(t+1)} = s, \mathbf{o}^{(1:t)}) p_{\mathbf{Y}^{(t+1)}}(\mathbf{y} \mid s^{(1:t)}(j), S^{(t+1)} = s, \mathbf{o}^{(1:t)}) \\
&= p(\mathbf{o}^{(t+1)} \mid s^{(1:t)}(j), S^{(t+1)} = s, \mathbf{o}^{(1:t)}) p_{\mathbf{Y}^{(t+1)}}(\mathbf{y} \mid \mathbf{o}^{(1:(t+1))}, s^{(1:t)}(j), S^{(t+1)} = s) \\
&= f(\mathbf{o}^{(t+1)}; \mathbf{C}_s \boldsymbol{\mu}_{j|s}^{(\cdot t+1)} + \mathbf{d}_s, \mathbf{R}_s + \mathbf{C}_s \boldsymbol{\Sigma}_{j|s}^{(\cdot t+1)} \mathbf{C}_s^{\mathsf{T}}) f(\mathbf{y}; \boldsymbol{\mu}_{j|s}^{(t+1)}, \boldsymbol{\Sigma}_{j|s}^{(t+1)}),
\end{aligned}$$

where we in the second equality used the conditional independence in Figure 3.3b, and in the third equality we used the result from Expression (3.19), (3.17) and (3.18).

We can find $\sigma^{(t+1)}(s) = p_s^{(t+1)}$ in a similar way as we found $p_s^{(\cdot t+1)}$, by integrating out $\mathbf{Y}^{(t+1)}$. Now we include the normalizing constant, which is given by Expression (3.6).

$$\begin{aligned}\sigma^{(t+1)}(s) = \int_{\mathbb{R}^N} \sigma^{(t+1)}(s,\mathbf{y})\,\mathrm{d}\mathbf{y} &= \int_{\mathbb{R}^N} \frac{p_s^{(\cdot t+1)} \sum_{j=1}^{M^t} \widetilde{v}_{j|s}^{(t+1)} f(\mathbf{y}; \boldsymbol{\mu}_{j|s}^{(t+1)}, \boldsymbol{\Sigma}_{j|s}^{(t+1)})}{p(\mathbf{o}^{(t+1)} \mid \mathbf{o}^{(1:t)})}\,\mathrm{d}\mathbf{y} \\ &= \frac{p_s^{(\cdot t+1)} \sum_{j=1}^{M^t} \widetilde{v}_{j|s}^{(t+1)}}{p(\mathbf{o}^{(t+1)} \mid \mathbf{o}^{(1:t)})}.\end{aligned}$$

Since the denominator is independent of $S^{(t+1)}$, we can write this as

$$p_s^{(t+1)} = \sigma^{(t+1)}(s) = \frac{\widetilde{p}_s^{(t+1)}}{\sum_{m=1}^{M} \widetilde{p}_m^{(t+1)}}, \tag{3.44}$$

where $\widetilde{p}_s^{(t+1)} = p_s^{(\cdot t+1)} \sum_{j=1}^{M^t} \widetilde{v}_{j|s}^{(t+1)}$ and $p(\mathbf{o}^{(t+1)} \mid \mathbf{o}^{(1:t)}) = \sum_{m=1}^{M} \widetilde{p}_m^{(t+1)}$. The weights $v_{j|s}^{(t+1)}$ are found by conditioning on $S^{(t+1)} = s$:

$$\begin{aligned}\sigma^{(t+1)}(\mathbf{y} \mid s) &= \frac{\sigma^{(t+1)}(s,\mathbf{y})}{\sigma^{(t+1)}(s)} \\ &= \frac{p_s^{(\cdot t+1)} \sum_{j=1}^{M^t} \widetilde{v}_{j|s}^{(t+1)} f(\mathbf{y}; \boldsymbol{\mu}_{j|s}^{(t+1)}, \boldsymbol{\Sigma}_{j|s}^{(t+1)}) / p(\mathbf{o}^{(t+1)} \mid \mathbf{o}^{(1:t)})}{p_s^{(t+1)}} \\ &= \frac{p_s^{(\cdot t+1)} \sum_{j=1}^{M^t} \widetilde{v}_{j|s}^{(t+1)} f(\mathbf{y}; \boldsymbol{\mu}_{j|s}^{(t+1)}, \boldsymbol{\Sigma}_{j|s}^{(t+1)})}{\widetilde{p}_s^{(t+1)}} \\ &= \frac{p_s^{(\cdot t+1)} \sum_{j=1}^{M^t} \widetilde{v}_{j|s}^{(t+1)} f(\mathbf{y}; \boldsymbol{\mu}_{j|s}^{(t+1)}, \boldsymbol{\Sigma}_{j|s}^{(t+1)})}{p_s^{(\cdot t+1)} \sum_{i=1}^{M^t} \widetilde{v}_{i|s}^{(t+1)}} \\ &= \sum_{j=1}^{M^t} \left(\frac{\widetilde{v}_{j|s}^{(t+1)}}{\sum_{i=1}^{M^t} \widetilde{v}_{i|s}^{(t+1)}} \right) f(\mathbf{y}; \boldsymbol{\mu}_{j|s}^{(t+1)}, \boldsymbol{\Sigma}_{j|s}^{(t+1)}),\end{aligned}$$

that is,

$$v_{j|s}^{(t+1)} = \frac{\widetilde{v}_{j|s}^{(t+1)}}{\sum_{i=1}^{M^t} \widetilde{v}_{i|s}^{(t+1)}}, \quad s = 1,\dots,M, \quad j = 1,\dots,M^t. \tag{3.45}$$

The complete procedure is shown in Algorithm 1, where

$$\begin{aligned}&\mathbf{A} = \{\mathbf{A}_s : s = 1,\dots,M\}, \quad \mathbf{b} = \{\mathbf{b}_s : s = 1,\dots,M\}, \quad \mathbf{Q} = \{\mathbf{Q}_s : s = 1,\dots,M\}, \\ &\mathbf{C} = \{\mathbf{C}_s : s = 1,\dots,M\}, \quad \mathbf{d} = \{\mathbf{d}_s : s = 1,\dots,M\}, \quad \mathbf{R} = \{\mathbf{R}_s : s = 1,\dots,M\}, \\ &\boldsymbol{\nu} = \{\boldsymbol{\nu}_s : s = 1,\dots,M\}, \quad \boldsymbol{\Gamma} = \{\boldsymbol{\Gamma}_s : s = 1,\dots,M\},\end{aligned} \tag{3.46}$$

and

$$\mathbf{p}^{(*)} = (p_s^{(*)}) \in \mathbb{R}^M, \quad (\mathbf{w}^{(*)}, \boldsymbol{\mu}^{(*)}, \boldsymbol{\Sigma}^{(*)}) = \{(w_k^{(*)}, \boldsymbol{\mu}_k^{(*)}, \boldsymbol{\Sigma}_k^{(*)}) : k = 1, \dots, M^t\}, \tag{3.47}$$

for $(*) = (\cdot t)$ and (t). In the prediction step of Algorithm 1, when we compute the parameters from $\sigma^{(\cdot t)}(s, \mathbf{y})$, the most expensive computation is forming $\boldsymbol{\Sigma}_k^{(\cdot t)}$ for each possible history $s^{(1:t)}(k)$, as we are dealing with a matrix-matrix product ($\mathcal{O}(N^3)$) in line 21 of the algorithm. This gives a complexity of $\mathcal{O}(M^t N^3)$. In the condition step of Algorithm 1, when we compute the parameters from $\sigma^{(t)}(s, \mathbf{y})$, the most expensive computation is either when we compute the Kalman gain $\mathbf{K}_k^{(t)}$ on line 31, since we have to compute the inverse of an $L \times L$ matrix ($\mathcal{O}(L^3)$), or when we compute $\boldsymbol{\Sigma}_k^{(t)}$ on line 33 where we have a matrix-matrix product ($\mathcal{O}(N^3)$), depending on which of M and N is largest. This gives a complexity of $\mathcal{O}(M^t \max\{N, L\}^3)$. In total, computing all the filtered estimates using the exact method, requires $\mathcal{O}(tM^t \max\{N, L\}^3)$ number of operations. It is clear we must find an alternative approximate method which does not grow exponentially to be able to perform inference on any observation sequence larger than $t = 15$.

Algorithm 1 Filter algorithm for SDLS

```
 1: procedure FILTER(
        $\theta = \{\mathbf{P}, \mathbf{A}, \mathbf{b}, \mathbf{Q}, \mathbf{C}, \mathbf{d}, \mathbf{R}, \boldsymbol{\pi}, \boldsymbol{\nu}, \boldsymbol{\Gamma}\}$        ▷ Model parameters
        $\mathbf{o}^{(1:T)}$                                         ▷ Observation sequence
    )
 2:     $\theta_H \leftarrow \{\mathbf{P}, \mathbf{A}, \mathbf{b}, \mathbf{Q}\}$            ▷ Model parameters for hidden variables
 3:     $\theta_O \leftarrow \{\mathbf{C}, \mathbf{d}, \mathbf{R}\}$                 ▷ Model parameters for observed variable
 4:     $\Theta^{(\cdot 1)} \leftarrow \{\boldsymbol{\pi}, 1, \boldsymbol{\nu}, \boldsymbol{\Gamma}\}$                  ▷ Prior belief state at time 1
 5:     $\Theta^{(1)} \leftarrow$ CONDSTEP($\theta_O, \Theta^{(\cdot 1)}, \mathbf{y}^{(1)}$)        ▷ Belief state at time 1
 6:     for $t = 1, \dots, T-1$ do
 7:         $\Theta^{(\cdot t+1)} \leftarrow$ PREDSTEP($\theta_H, \Theta^{(t)}$)                ▷ Prior belief state
 8:         $\Theta^{(t+1)} \leftarrow$ CONDSTEP($\theta_O, \Theta^{(\cdot t+1)}, \mathbf{y}^{(t+1)}$)      ▷ Belief state
 9:     end for
10:     return $\Theta^{(1:T)}$
11: end procedure
```

3.4.3 Approximate deterministic inference

It is clear from Expression (3.26) that computing the exact filtering estimates as described in Algorithm 1 is not possible, because of the exponentially growing mixture.

A natural suggestion would be to restrict the Gaussian mixture to a fixed amount of Gaussian densities each time step. There are many possibilities when choosing how large we wish the mixture to be, but typically it would consist of $\min\{M^{p-1}, M^t\}$ Gaussian densities where p is a positive integer, since the mixture grows by a factor of M Gaussian densities per time step. The larger we choose p, the more accurate the mixture would

```
12: procedure PredStep(
        $\theta_H = \{\mathbf{P}, \mathbf{A}, \mathbf{b}, \mathbf{Q}\}$        ▷ Model parameters for hidden variables
        $\Theta^{(t)} = \{\mathbf{p}^{(t)}, \mathbf{w}^{(t)}, \boldsymbol{\mu}^{(t)}, \boldsymbol{\Sigma}^{(t)}\}$        ▷ Prior belief state
    )
13:     for $s = 1, \ldots, M$ do
14:         $p_s^{(\cdot t+1)} \leftarrow \sum_{m=1}^{M} p_{ms} p_m^{(t)}$        ▷ $P(S^{(t+1)} = s \mid \mathbf{o}^{(1:t)})$
15:         for $j = 1 \ldots, M^t$ do
16:             $k \leftarrow (s-1)M^t + j$        ▷ Index for history $s^{(1:(t+1))}(k)$
17:             $m \leftarrow \left\lfloor \frac{j-1}{M^{t-1}} \right\rfloor + 1$        ▷ Index for $S^{(t)} = s^{(t)}(k) = s^{(t)}(j) = m$
18:             $w_k^{(\cdot t+1)} \leftarrow p_{ms} w_j^{(t)}$        ▷ Weight $P(s^{(1:(t+1))}(k) \mid \mathbf{o}^{(1:t)})$
19:             // Compute moments of the PDF $p(\mathbf{y}^{(t+1)} \mid s^{(1:(t+1))}(k), \mathbf{o}^{(1:t)})$
20:             $\boldsymbol{\mu}_k^{(\cdot t+1)} \leftarrow \mathbf{A}_s \boldsymbol{\mu}_j^{(t)} + \mathbf{b}_s$
21:             $\boldsymbol{\Sigma}_k^{(\cdot t+1)} \leftarrow \mathbf{Q}_s + \mathbf{A}_s \boldsymbol{\Sigma}_j^{(t)} \mathbf{A}_s^\intercal$
22:         end for
23:     end for
24:     return $\{\mathbf{p}^{(\cdot t+1)}, \mathbf{w}^{(\cdot t+1)}, \boldsymbol{\mu}^{(\cdot t+1)}, \boldsymbol{\Sigma}^{(\cdot t+1)}\}$
25: end procedure
```

be. Setting $p = 1$ gives us the most gross approximation where the filtered estimate for $\mathbf{Y}^{(t)}$ would be a single Gaussian.

One important question we must answer, is when we do the approximation. If we decided to have a mixture of M^{p-1} Gaussian densities, then at the prediction step, the mixture increases to M^p Gaussian densities. Should we approximate here, to go back to M^{p-1} Gaussian densities, or should we do the condition step first? If we choose to do the approximation in the prediction step, we may have a faster running code than if we choose to do the approximation in the condition step. However, we may lose some accuracy choosing to do the approximation before we have conditioned on the newest observation in the condition step.

Another very important question we must answer is *how* we approximate. How do we go from M^p Gaussian densities to M^{p-1} Gaussian densities? Assume that we choose to do the approximation after we have conditioned on the latest observation. Then we have to do the following approximation step

$$\breve{\sigma}^{(t)}(\mathbf{y}) = \sum_{k=1}^{M^p} \breve{w}_k^{(\cdot t)} f(\mathbf{y}; \breve{\boldsymbol{\mu}}_k^{(t)}, \breve{\boldsymbol{\Sigma}}_k^{(t)}) \approx \sum_{k=1}^{M^{p-1}} \widehat{w}_k^{(\cdot t)} f(\mathbf{y}; \widehat{\boldsymbol{\mu}}_k^{(t)}, \widehat{\boldsymbol{\Sigma}}_k^{(t)}) = \widehat{\sigma}^{(t)}(\mathbf{y}).$$

The most straightforward way is to discard the $M^{p-1}(M-1)$ lowest weights and normalize the remaining weights. This method is called pruning. However, eventually it is unavoidable to discard a region where the true $Y^{(t)}$ had taken place. Although there is a low probability of $Y^{(t)}$ taking a value in this region, it is still possible.

Another more promising method, is to collapse the mixture while conserving the first

```
26: procedure CondStep(
        $\theta_O = \{\mathbf{C}, \mathbf{d}, \mathbf{R}\}$    ▷ Model parameters for observed variable
        $\Theta^{(\cdot t+1)} = \{\mathbf{p}^{(\cdot t+1)}, \mathbf{w}^{(\cdot t+1)}, \boldsymbol{\mu}^{(\cdot t+1)}, \boldsymbol{\Sigma}^{(\cdot t+1)}\}$    ▷ Predicted belief state
        $\mathbf{o}^{(t+1)}$    ▷ Time $t+1$ observation
    )
27:     for $s = 1, \dots, M$ do
28:         for $j = 1, \dots, M^t$ do
29:             $k \leftarrow (s-1)M^t + j$    ▷ Index for history $s^{(1:(t+1))}(k)$
30:             // Compute moments of the PDF $p(\mathbf{y}^{(t+1)} \mid s^{(1:(t+1))}(k), \mathbf{o}^{(1:(t+1))})$
31:             $\mathbf{K}_k^{(t+1)} \leftarrow \boldsymbol{\Sigma}_k^{(\cdot t+1)} \mathbf{C}_s^\mathsf{T} (\mathbf{R}_s + \mathbf{C}_s \boldsymbol{\Sigma}_k^{(\cdot t+1)} \mathbf{C}_s^\mathsf{T})^{-1}$    ▷ Kalman gain
32:             $\boldsymbol{\mu}_k^{(t+1)} \leftarrow \boldsymbol{\mu}_k^{(\cdot t+1)} + \mathbf{K}_k^{(t+1)}(\mathbf{o}^{(t+1)} - (\mathbf{C}_s \boldsymbol{\mu}_k^{(\cdot t+1)} + \mathbf{d}_s))$
33:             $\boldsymbol{\Sigma}_k^{(t+1)} \leftarrow (\mathbf{I} - \mathbf{K}_k^{(t+1)} \mathbf{C}_s) \boldsymbol{\Sigma}_k^{(\cdot t+1)}$
34:             // Compute $\mathcal{P}(s^{(1:t)}(j), \mathbf{o}^{(t+1)} \mid S^{(t+1)} = s, \mathbf{o}^{(1:t)})$
35:             $\widetilde{v}_{j|s}^{(t+1)} \leftarrow \frac{w_k^{(\cdot t+1)}}{p_s^{(\cdot t+1)}} f(\mathbf{o}^{(t+1)} \mid \mathbf{C}_s \boldsymbol{\mu}_k^{(\cdot t+1)} + \mathbf{d}_s, \mathbf{R}_s + \mathbf{C}_s \boldsymbol{\Sigma}_k^{(\cdot t+1)} \mathbf{C}_s^\mathsf{T})$
36:         end for
37:         $z_s^{(t+1)} \leftarrow \sum_{j=1}^{M^t} \widetilde{v}_{j|s}^{(t+1)}$    ▷ $p(\mathbf{o}^{(t+1)} \mid S^{(t+1)} = s, \mathbf{o}^{(1:t)})$
38:         $\widetilde{p}_s^{(t+1)} \leftarrow p_s^{(\cdot t+1)} z_s^{(t+1)}$    ▷ $\mathcal{P}(S^{(t+1)} = s, \mathbf{o}^{(t+1)} \mid \mathbf{o}^{(1:t)})$
39:     end for
40:     $L^{(t+1)} \leftarrow \sum_{j=1}^{M} \widetilde{p}_j^{(t+1)}$    ▷ Conditional likelihood $p(\mathbf{o}^{(t+1)} \mid \mathbf{o}^{(1:t)})$
41:     $\ell^{(t+1)} \leftarrow \log L^{(t+1)}$    ▷ Log-likelihood
42:     for $s = 1, \dots, M$ do
43:         $p_s^{(t+1)} \leftarrow \widetilde{p}_s^{(t+1)} / L^{(t+1)}$    ▷ $P(S^{(t+1)} = s \mid \mathbf{o}^{(1:(t+1))})$
44:         for $j = 1, \dots, M^t$ do
45:             $k \leftarrow (s-1)M^t + j$    ▷ Index for history $s^{(1:(t+1))}(k)$
46:             $v_{j|s}^{(t+1)} \leftarrow \widetilde{v}_{j|s}^{(t+1)} / z_s^{(t+1)}$    ▷ Weight $P(s^{(1:t)}(j) \mid S^{(t+1)} = s, \mathbf{o}^{(1:(t+1))})$
47:             $w_k^{(t+1)} \leftarrow p_s^{(t+1)} v_{j|s}^{(t+1)}$    ▷ Weight $P(s^{(1:(t+1))}(k) \mid \mathbf{o}^{(1:(t+1))})$
48:         end for
49:     end for
50:     return $\{\mathbf{p}^{(t+1)}, \mathbf{w}^{(t+1)}, \boldsymbol{\mu}^{(t+1)}, \boldsymbol{\Sigma}^{(t+1)}, \ell^{(t+1)}\}$
51: end procedure
```

two moments (mean and variance). If we have a Gaussian mixture

$$p(\mathbf{y}) = \sum_{i=1}^{K} w_i f(\mathbf{y}; \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i),$$

then the moment matched density $\widehat{p}(\mathbf{y}) = f(\mathbf{y}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$ has mean and variance

$$\begin{aligned} \boldsymbol{\mu} &= \sum_{i=1}^{K} w_i \boldsymbol{\mu}_i, \\ \boldsymbol{\Sigma} &= \sum_{i=1}^{K} w_i \boldsymbol{\Sigma}_i + \sum_{i=1}^{K} w_i (\boldsymbol{\mu}_i - \boldsymbol{\mu})(\boldsymbol{\mu}_i - \boldsymbol{\mu})^{\mathsf{T}}. \end{aligned} \tag{3.48}$$

The first line in Expression (3.48) seems intuitive as it is simply the weighted average of each individual mean from the mixture. The second line also has a weighted average of the variances, but has an additional term that takes into account the "distance" between each individual density. We see that the larger the distance between the mean $\boldsymbol{\mu}$ and the individual means $\boldsymbol{\mu}_i$, the larger the extra term becomes.

An example of the two methods is shown in Figure 3.7, where we have a mixture of two Gaussian densities, $f(y; 3, 1^2)$ and $f(y; 6, 2^2)$, with the weights 0.7 and 0.3, respectively. To the left, we see that the mixture is approximated by discarding the density $f(y; 3, 1^2)$ because it had the lowest weight. We immediately see that the probability of Y being greater than 6 is reduced by a lot after the approximation is made, whereas the moment matching approximation appears to give a more similar probability of Y being greater than 6. The probabilities are as follows:

$$\begin{aligned} &P(Y > 6) = 0.1441 \quad \text{(Original mix)}, \\ &P(Y > 6) = 0.0013 \quad \text{(Collapse by pruning)}, \\ &P(Y > 6) = 0.1164 \quad \text{(Collapse by moment matching)}. \end{aligned}$$

In this case, choosing the moment matching method seems to be the better choice, and this is true in most cases. However, there may be cases where the mixture consists of very distinct densities as in Figure 3.8, representing different hypotheses, and we wish to determine which hypothesis seems more probable. By pruning, we reject that Y lies in the neighborhood around 60, but if we collapse by moment matching, we get a bit of both hypotheses. However, the most likely value Y can take in the moment matched approximation is $Y = 39$, that lies in a region of which the original mixture has a extremely low probability. Lerner (2003) proposes an algorithm that combines the pruning method and the moment matching method having a tuning parameter. However, we choose to continue with the moment matching method as it often works great in practice.

Collapsing by moment matching to maintain a mixture of M^{p-1} Gaussian densities in each time step is called *Generalized Pseudo Bayesian p*, or GPB(p) (Blom & Bar-Shalom, 1988; Murphy, 2002; Lerner, 2003; Koller & Friedman, 2009). The most

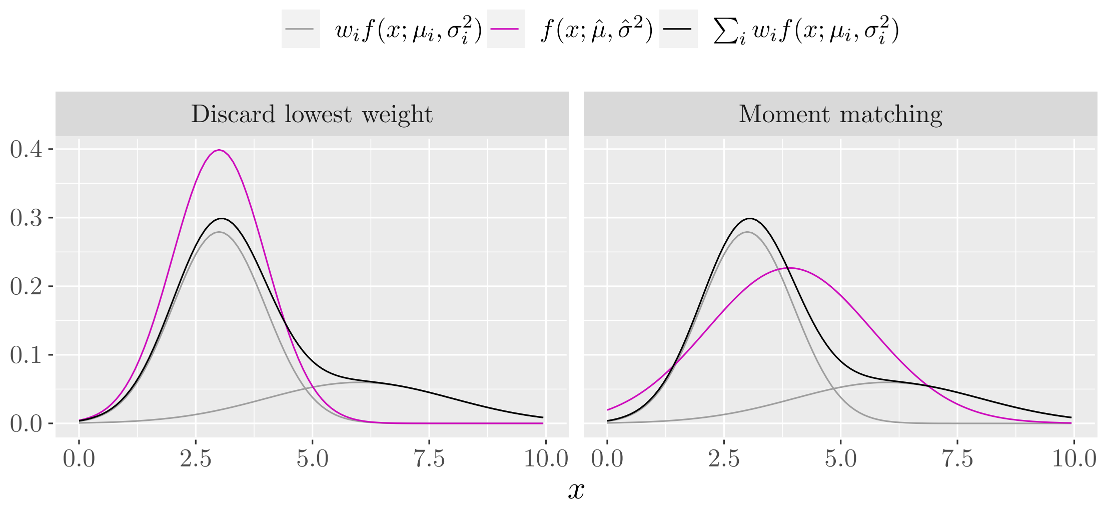


Figure 3.7: Two approximations of a Gaussian mixture, where the gray curves are the mixture components, the black curves are the mixture and the purple curves are the collapsed approximations. The approximation to the left is made by choosing the Gaussian density with the highest weight. The approximation to the right is done by choosing a single Gaussian density with the same mean and variance as the mixture.

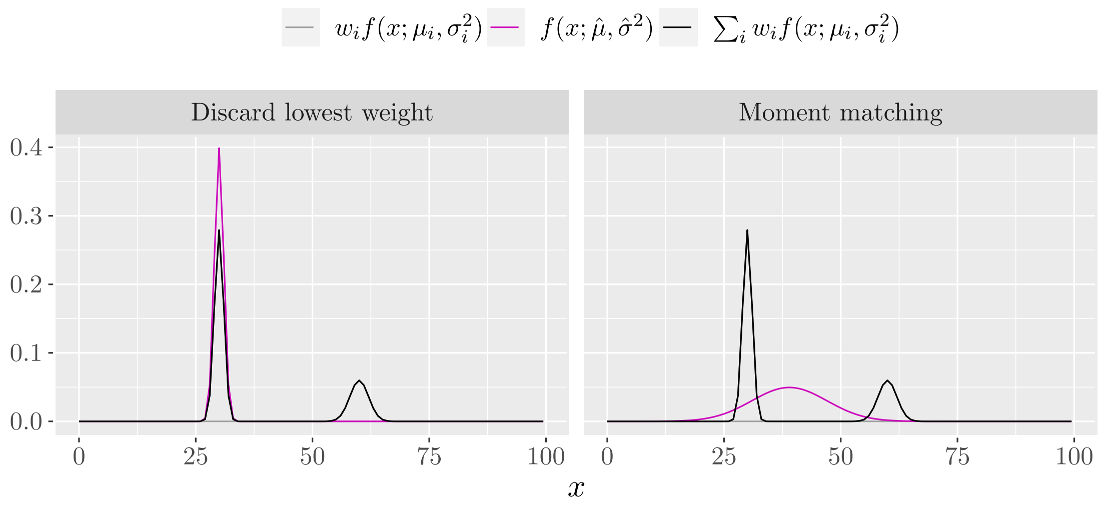


Figure 3.8: Another example of two approximations of a Gaussian mixture.

common approximations are choosing $p = 1$ and 2, giving them the names GPB1 and GPB2. A schematic of the two methods are shown in Figure 3.9.

$$\begin{tikzcd}
 & \{p_s^{(t)}\}_{s=1}^M, \boldsymbol{\mu}^{(t)}, \boldsymbol{\Sigma}^{(t)} \arrow[d] \\
 & \boxed{\text{Predict}} \arrow[d] \\
 & \{p_s^{(\cdot t+1)}, \boldsymbol{\mu}_s^{(\cdot t+1)}, \boldsymbol{\Sigma}_s^{(\cdot t+1)}\}_{s=1}^M \arrow[d] \\
\mathbf{o}^{(t+1)} \arrow[r] & \boxed{\text{Condition}} \arrow[d] \\
 & \{p_s^{(t+1)}, \widehat{\boldsymbol{\mu}}_s^{(t+1)}, \widehat{\boldsymbol{\Sigma}}_s^{(t+1)}\}_{s=1}^M \arrow[d] \\
 & \boxed{\text{Collapse}} \arrow[d] \\
 & \{p_s^{(t+1)}\}_{s=1}^M, \boldsymbol{\mu}^{(t+1)}, \boldsymbol{\Sigma}^{(t+1)}
\end{tikzcd}$$

(a) GPB1 $\mathcal{O}(M)$.

$$\begin{tikzcd}
 & \{p_s^{(t)}, \boldsymbol{\mu}_s^{(t)}, \boldsymbol{\Sigma}_s^{(t)}\}_{s=1}^M \arrow[d] \\
 & \boxed{\text{Predict}} \arrow[d] \\
 & \{p_s^{(\cdot t+1)}, \{v_{j|s}^{(\cdot t+1)}, \boldsymbol{\mu}_{j|s}^{(\cdot t+1)}, \boldsymbol{\Sigma}_{j|s}^{(\cdot t+1)}\}_{j=1}^M\}_{s=1}^M \arrow[d] \\
\mathbf{o}^{(t+1)} \arrow[r] & \boxed{\text{Condition}} \arrow[d] \\
 & \{p_s^{(t+1)}, \{v_{j|s}^{(t+1)}, \boldsymbol{\mu}_{j|s}^{(t+1)}, \boldsymbol{\Sigma}_{j|s}^{(t+1)}\}_{j=1}^M\}_{s=1}^M \arrow[d] \\
 & \boxed{\text{Collapse}} \arrow[d] \\
 & \{p_s^{(t+1)}, \boldsymbol{\mu}_s^{(t+1)}, \boldsymbol{\Sigma}_s^{(t+1)}\}_{s=1}^M
\end{tikzcd}$$

(b) GPB2 $\mathcal{O}(M^2)$.

Figure 3.9: The schematic shows one cycle of the filtering steps using the Generalized Pseudo Bayesian approximations GPB1 and GPB2. The collapse happens after the condition step.

The simplest one, the GPB1, starts with a single Gaussian density. Then in the prediction step, it increases to a mixture of M Gaussians densities having a computational cost of $\mathcal{O}(MN^3)$. Then in the condition step the computational cost is similar, depending on whether N or L is largest, giving a cost of $\mathcal{O}(M \max\{N, L\}^3)$. We currently have a mixture of M Gaussians which we need to collapse. As we are collapsing to a single Gaussian, this is straight forward, where we moment project directly using Expression (3.48). This has a cost of $\mathcal{O}(MN^2)$ because of the summation of the variance terms. Thus, the GPB1 method has a total cost of $\mathcal{O}(tM \max\{N, L\}^3)$. It is now linear in terms of the number of classes M, compared to the exponential M^t for exact inference.

The GPB2 starts off similar to GPB1, but with a mixture of M Gaussians instead. This mixture increases to M^2 Gaussians which we have to collapse in after the condition step. We have to collapse the mixture from M^2 Gaussians to M Gaussians. It seems that we cannot moment project this using Expression (3.48), since that would give us a single Gaussian. The trick is to use the alternative indexing, shown in Expression (3.29), where we condition on the most recent $S^{(t)}$. Here we must use the conditioned weights $v_{j|s}^{(t)}$ instead of $w_k^{(t)}$ in the collapse, and moment project as follows:

$$\sum_{j=1}^{M} v_{j|s}^{(t)} f(\mathbf{y}; \boldsymbol{\mu}_{j|s}^{(t)}, \boldsymbol{\Sigma}_{j|s}^{(t)}) \approx f(\mathbf{y}; \boldsymbol{\mu}_s^{(t)}, \boldsymbol{\Sigma}_s^{(t)}), \quad \text{for} \quad s = 1, \ldots, M.$$

$$\begin{tikzcd}
 & \{p_s^{(t)}, \boldsymbol{\mu}_s^{(t)}, \boldsymbol{\Sigma}_s^{(t)}\}_{s=1}^M \arrow[d] \\
 & \boxed{\text{Predict pt. 1}} \arrow[d] \\
 & \{p_s^{(\cdot t+1)}, \{v_{j|s}^{(\cdot t+1)}, \boldsymbol{\mu}_j^{(t)}, \boldsymbol{\Sigma}_j^{(t)}\}_{j=1}^M\}_{s=1}^M \arrow[d] \\
 & \boxed{\text{Collapse}} \arrow[d] \\
 & \{p_s^{(\cdot t+1)}, \breve{\boldsymbol{\mu}}_s^{(\cdot t+1)}, \breve{\boldsymbol{\Sigma}}_s^{(\cdot t+1)}\}_{s=1}^M \arrow[d] \\
 & \boxed{\text{Predict pt. 2}} \arrow[d] \\
 & \{p_s^{(\cdot t+1)}, \boldsymbol{\mu}_s^{(\cdot t+1)}, \boldsymbol{\Sigma}_s^{(\cdot t+1)}\}_{s=1}^M \arrow[d] \\
\mathbf{o}^{(t+1)} \arrow[r] & \boxed{\text{Condition}} \arrow[d] \\
 & \{p_s^{(t+1)}, \boldsymbol{\mu}_s^{(t+1)}, \boldsymbol{\Sigma}_s^{(t+1)}\}_{s=1}^M
\end{tikzcd}$$

Figure 3.10: The schematic shows one cycle of the filtering steps using the IMM approach, $(\mathcal{O}(M))$.

The GPB2 method is closer to the exact solution but the cost has increased by a factor of M compared to GPB1, that is, it is quadratic in terms of M. This leaves us with a trade-off between computational performance and accuracy of the approximation. For large M, GPB2 may be too computationally demanding, forcing us to go for the more gross approximation of GPB1. What if both GPB2 is too expensive, and GPB1 gives a too poor filtered estimate?

There is a third way, where we get the best of both methods. That is, we get computational cost which is linear in M as in GPB1, but we get a Gaussian mixture of M Gaussian densities as in GPB2. This method is called *Interactive Multiple Models* or IMM (Blom & Bar-Shalom, 1988). Here we do the collapsing step in the beginning of the prediction step, instead of after the condition step. At first, this may seem like we are doing the GPB1 approach, with the collapse in the beginning instead of the end, but there is a subtle difference. A schematic is shown in Figure 3.10. In the prediction step, we first compute the conditioned weights $\{v_{j|s}^{(\cdot t+1)}\}_{j=1}^M$ for $s = 1, \dots, M$. That is, we get a unique set of weights for each s, that we use to collapse the filtered parameters

$\boldsymbol{\mu}_k^{(t)}, \boldsymbol{\Sigma}_k^{(t)}$:

$$\sum_{j=1}^{M} v_{j|s}^{(\cdot t+1)} f(\mathbf{y}; \boldsymbol{\mu}_j^{(t)}, \boldsymbol{\Sigma}_j^{(t)}) \approx f(\mathbf{y}; \breve{\boldsymbol{\mu}}_s^{(\cdot t+1)}, \breve{\boldsymbol{\Sigma}}_s^{(\cdot t+1)}) \quad \text{for} \quad s = 1, \ldots, M.$$

That is, we use the same set of M Gaussians for each s, giving us a new set of M Gaussians, because of the different weights $v_{j|s}$. To finish off the prediction step, we compute

$$\begin{aligned} \boldsymbol{\mu}_s^{(\cdot t+1)} &= \mathbf{A}_s \breve{\boldsymbol{\mu}}_s^{(\cdot t+1)} + \mathbf{b}_s, \\ \boldsymbol{\Sigma}_s^{(\cdot t+1)} &= \mathbf{Q}_s + \mathbf{A}_s \breve{\boldsymbol{\Sigma}}_s^{(\cdot t+1)} \mathbf{A}_s^{\mathsf{T}}, \end{aligned} \qquad s = 1, \ldots, M,$$

which if you compare to Algorithm 1 on lines 20 and 21, we see that in IMM, we do the computation for each $s = 1, \ldots, M$, instead of for each $s, j = 1, \ldots, M$ as in GPB2, saving us from having a quadratic cost in M.

3.4.4 Approximate stochastic inference - Particle filtering

As we have seen, doing inference on SLDSs requires a lot of analytical computation beforehand and it is time consuming to implement the algorithms. This motivates alternative stochastic approaches. Stochastic approaches are in general very flexible in terms of the model, and often requires little to no analytical computation beforehand. This is also the case for the stochastic method we show in this section, the *particle filtering* algorithm (Murphy, 2002; Lerner, 2003; Koller & Friedman, 2009). There are many variants of the particle filter, but the simplest and the most common one is the likelihood weighting, which we now show.

The SLDSs assumes that the continuous part of the transition model $\mathbf{Y}' \mid \{S', \mathbf{Y}\}$ is linear Gaussian, similarly for for the observation model $\mathbf{Y} \mid \{S, \mathbf{Y}\}$, but the particle filtering algorithm can handle non-linearity. The basic idea of the particle filter is shown in Figure 3.11. We assume we have B samples $\{s^{(t)}[b], \mathbf{y}^{(t)}[b]\}_{b=1}^{B}$ which we call particles, that are drawn from the filtered estimate $\sigma^{(t)}(s, \mathbf{y})$. Then, for each particle $(s^{(t)}[b], \mathbf{y}^{(t)}[b])$, we draw one particle $(s^{(\cdot t+1)}[b], \mathbf{y}^{(\cdot t+1)}[b])$ from the transition model $\mathcal{P}(S', \mathbf{Y}' \mid S, \mathbf{Y})$, which gives us B new particles. We now have an unfiltered set of particles for time $t+1$, that are drawn from the predicted estimate $\sigma^{(\cdot t+1)}(s, \mathbf{y})$. These are the particles from the top row in Figure 3.11. The next step is to associate each particle with a weight $w^{(t+1)}[b]$ which indicates how likely it is to observe $\mathbf{o}^{(t+1)}$ when the hidden state is given by the current particle. That is,

$$w^{(t+1)}[b] = \mathcal{P}(o^{(t+1)} \mid \mathbf{y}^{(t+1)}[b], s^{(t+1)}[b]), \tag{3.49}$$

which is shown in the middle row in Figure 3.11. Thus, we have a way of ranking the particles based on how "good" they are. That is, particles that receive a high weight are particles that explain the evidence ($\mathbf{o}^{(t+1)}$) better. We then bootstrap the indices $b = 1, \ldots, B$ using the the weights, to select the most likely particles at time $t+1$,

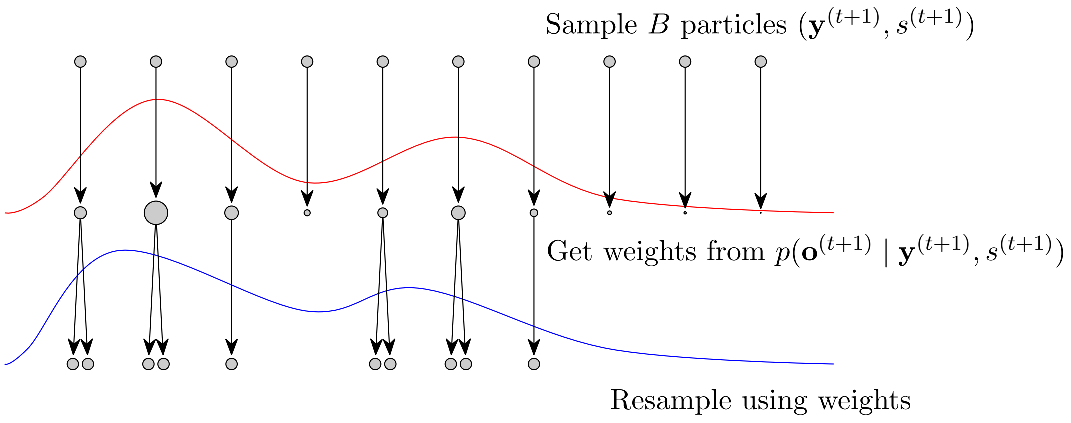

Sample B particles $(\mathbf{y}^{(t+1)}, s^{(t+1)})$
Get weights from $p(\mathbf{o}^{(t+1)} \mid \mathbf{y}^{(t+1)}, s^{(t+1)})$
Resample using weights

Figure 3.11: Particle filtering $\mathcal{O}(B)$. The equal-sized circles in the top row represents the B particles sampled from the transition model. The particles then get a weight from inserting them into the observation model together with the observation $\mathbf{o}^{(t+1)}$, where the red curve represents the likelihood of getting the different particles.

which is shown in the bottom row in Figure 3.11. Then this cycle repeats. This gives a complexity of (B) for each time step, which for any reasonable size of B gives a higher running time than the deterministic approximations.

One problem with this simple approach, is that the particle randomly shoots out a new particle at the next time step without considering the previous observations, and if we are "lucky", the generated particle matches the observation by chance. However, we choose to give it a try in the upcoming example..

3.4.5 Comparison

The following example is inspired by Meuter et al. (2009). Bob is in his car, on his way to visit his friend Alice. Bob is currently the only car on the freeway so he decides to challenge Alice with a guessing game. He starts to share his location with Alice, such that she receives GPS data every second. The freeway has three lanes, and he wants her to use the GPS data to guess which lane he is switching to. Since the freeway is a straight line, Alice thinks this will be an easy task and accepts the challenge. First she transforms the GPS-coordinates to a value that determines the shortest (signed) distance between the car and the right shoulder of the freeway in meters. If the data point is on the right side of the right shoulder, we get a negative distance. The total road width is 10.5 meters, so each lane is 3.5 meters wide.

Alice sets up the following model: Let $S^{(t)} \in \{1, 2, 3\}$ be the lane Bob is driving on, where 1 is the lane next to the right shoulder, and 3 is the lane furthest away from the right shoulder. Let $Y^{(t)}$ be the distance between Bob's car and the right shoulder, and similarly let $O^{(t)}$ be the observed distance between Bob's car and the right shoulder. Bob tells Alice how long he expects to stay on each lane, so Alice sets the following parameters

$$\begin{gathered}
\boldsymbol{\pi} = \begin{bmatrix} 1/3 \\ 1/3 \\ 1/3 \end{bmatrix}, \quad \mathbf{P} = \begin{bmatrix} 0.99 & 0.01 & 0 \\ 0.01 & 0.98 & 0.01 \\ 0 & 0.01 & 0.99 \end{bmatrix}; \\
\nu_1 = 1.75, \quad \nu_2 = 5.25, \quad \nu_3 = 8.75, \quad \Gamma_1 = \Gamma_2 = \Gamma_3 = 1; \\
A_s = 0.8, \quad b_s = (1 - A_s)\nu_s, \quad Q_s = .02^2, \quad s = 1, 2, 3; \\
C = 1, \quad d = 0, \quad R = 2^2.
\end{gathered} \tag{3.50}$$

That is

$$\begin{aligned}
Y^{(1)} \mid \{S^{(t)} = s\} &\sim \mathcal{N}(\nu_s, \Gamma_s) \\
Y' \mid \{S' = s', Y = y\} &\sim \mathcal{N}(A_{s'}y + b_{s'}, Q_{s'}) \\
O \mid \{Y = y\} &\sim \mathcal{N}(Cy + d, R) = \mathcal{N}(y, 2^2).
\end{aligned} \tag{3.51}$$

The reason for choosing $b_s = (1 - A_s)\nu_s$, is that we want the limiting mean, $\mu^{(\infty)}$ to converge to ν_s, so we solve for b_s the following equation, (ignoring the switching state for the moment):

$$\begin{aligned}
\mu^{(\infty)} &= \lim_{t\to\infty} \mathrm{E}[Y^{(t+1)}] = \lim_{t\to\infty} \mathrm{E}[\mathrm{E}(Y^{(t+1)} \mid Y^{(t)})] = \lim_{t\to\infty} A\,\mathrm{E}[Y^{(t)}] + b \\
&= A\mu^{(\infty)} + b \\
\Rightarrow b &= (1 - A)\mu^{(\infty)}.
\end{aligned} \tag{3.52}$$

Figure 3.12 shows Bob's driving pattern for 200 seconds. He starts at the third lane for about fifty seconds before he suddenly switches to the second lane, and finally he switches to the first lane. We also see the data points that Alice receives from Bob (crosses).

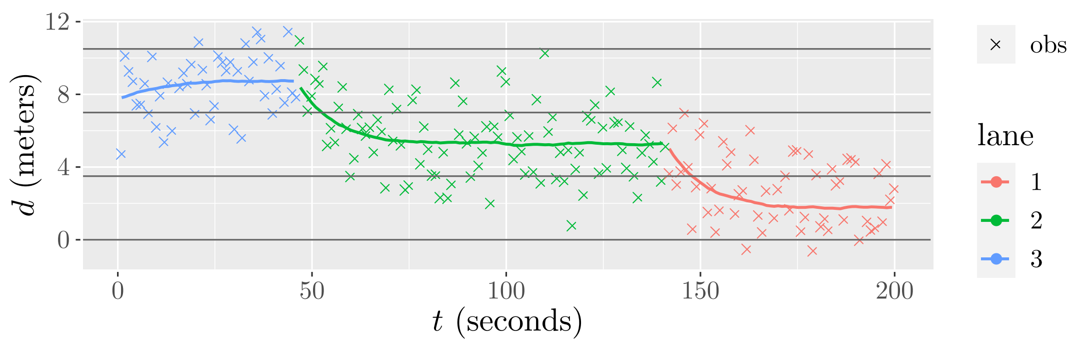
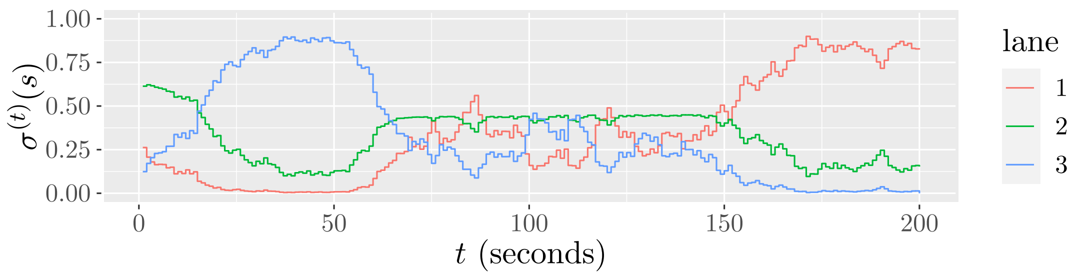
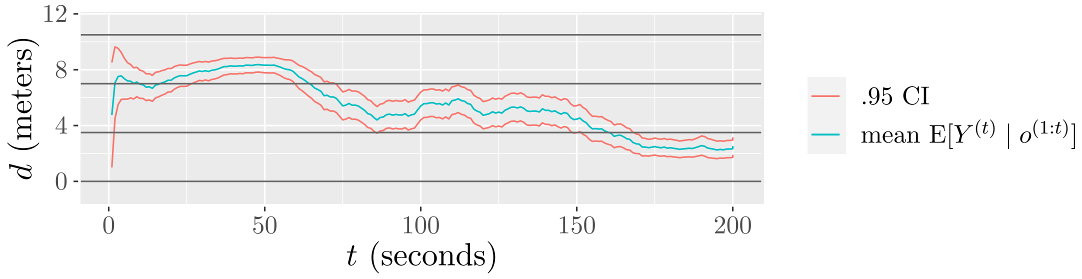


Figure 3.12: The true (solid line) and measured (crossed points) displacement from the right shoulder of the freeway. The three lanes are separated by the black horizontal lines.

Alice knows that 200 data points is too much to do exact inference, so she decides to try out GPB1, GPB2, IMM as well as particle filtering. The results are shown in

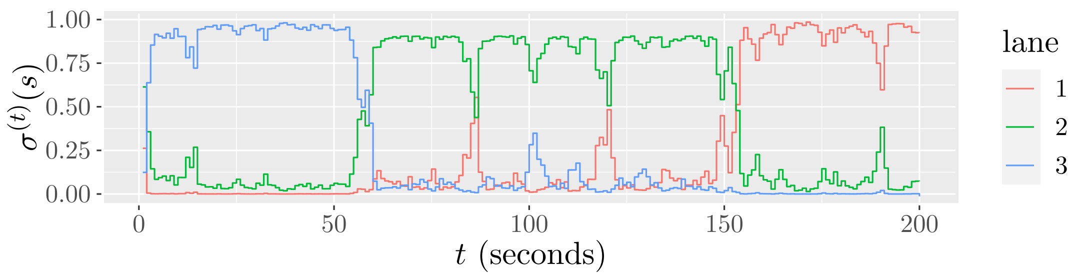


(a) Filtered estimate of the lane switches.

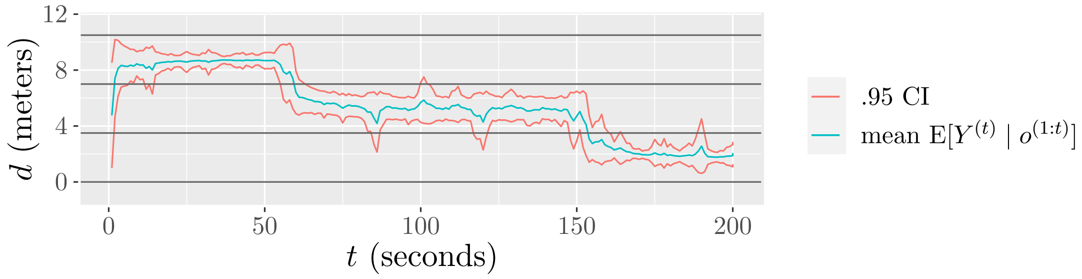


(b) Filtered estimate of the displacement from the right shoulder of the road.

Figure 3.13: Filtered estimate using GPB1.

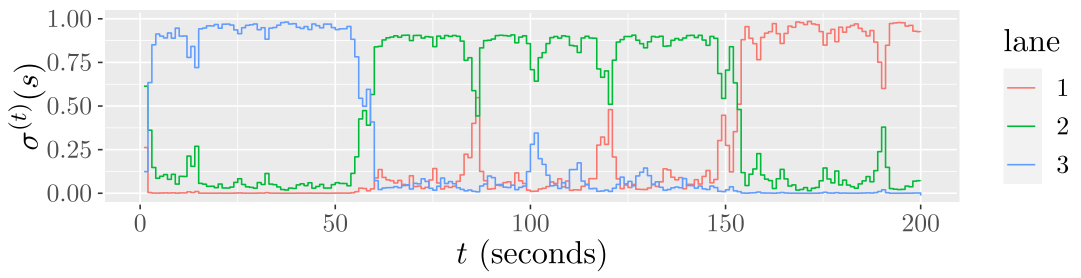


(a) Filtered estimate of the lane switches.

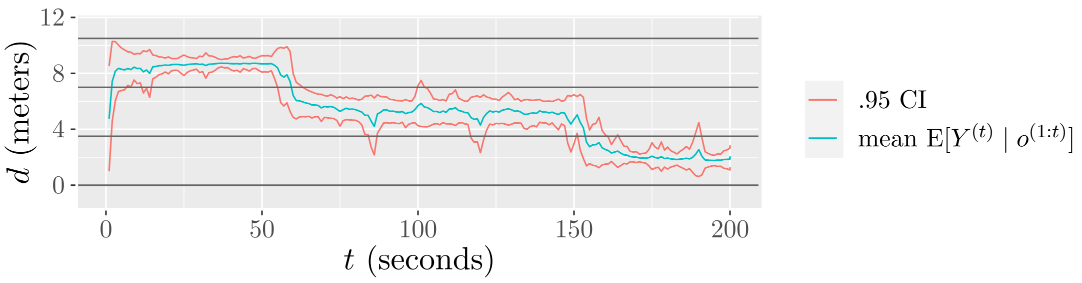


(b) Filtered estimate of the displacement from the right shoulder of the road.

Figure 3.14: Filtered estimate using GPB2.

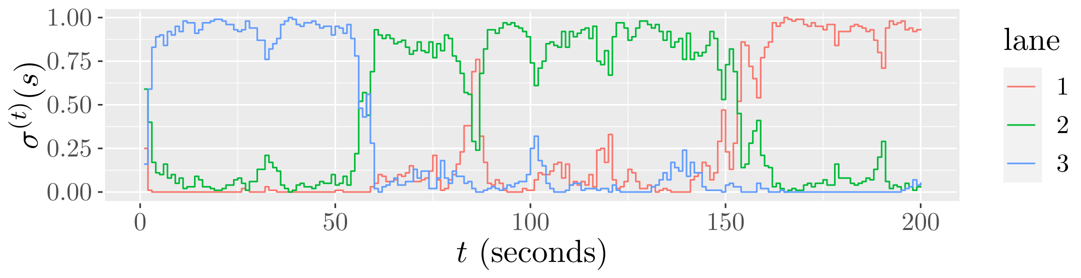


(a) Filtered estimate of the lane switches.

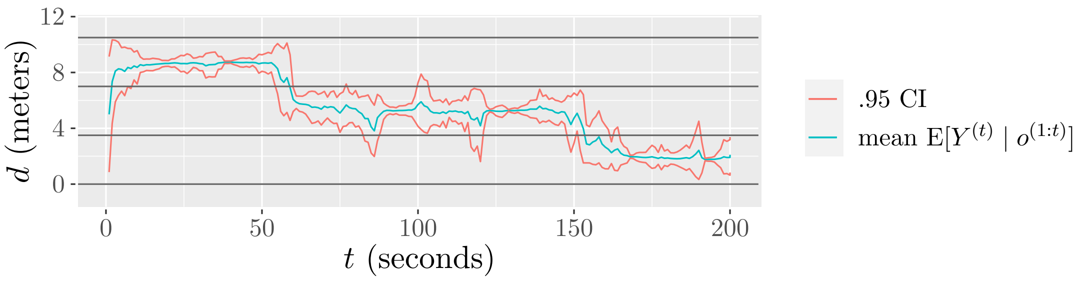


(b) Filtered estimate of the displacement from the right shoulder of the road.

Figure 3.15: Filtered estimate using IMM.



(a) Filtered estimate of the lane switches.



(b) Filtered estimate of the displacement from the right shoulder of the road.

Figure 3.16: Filtered estimate using particle filtering using $B = 100$ particles.

Figure 3.13 to 3.16. Alice sees that the results from GPB2, IMM and the particle filter all agrees that Bob is starting at lane 3, then switches to lane 2 at around sixty seconds, and then swaps to lane 1 at around 155 seconds, whereas the GPB1 approximation is a bit more uncertain, especially in the middle where all lanes are almost equally likely. The lane switches that Alice guesses are correct, but she detects the switches about 10-20 seconds late. This is due to the noisy GPS data from Bob's car. Had it been less noisy, the filtering algorithm would detect the switch sooner. Notice the 0.95 confidence interval in Figure 3.13b. It is much smoother in GPB1 than the others. This is explained by that the GPB1 always collapses to "one belief", whereas the others approximations are still considering other possibilities in the mixture.

We see that the approximations works well in this example, but how well? By using the same parameters as from the example, we generate 50 simulations for $t = 10$ time steps. The reason for this low number of time steps is so we can compare the approximations with the exact inference. Figure 3.17 shows the accuracy of the switching state (lane) for all approximations, together with the exact filtered estimate, as well as the root mean square error for the dynamic state (displacement from the right shoulder). From worst to best on accuracy, we have the particle filter for 50 particles at the gives the worst accuracy, which is followed by the particle filter for 200 particles. That is, all of the deterministic approaches surpasses the particle filter on accuracy. GPB1 is next on the list, and we see that GPB2 and IMM performs equally as well. When comparing with the exact approach, they overlap. This shows that due to the randomness in the process, the history only a few time steps back is not crucial for determining the correct current state.

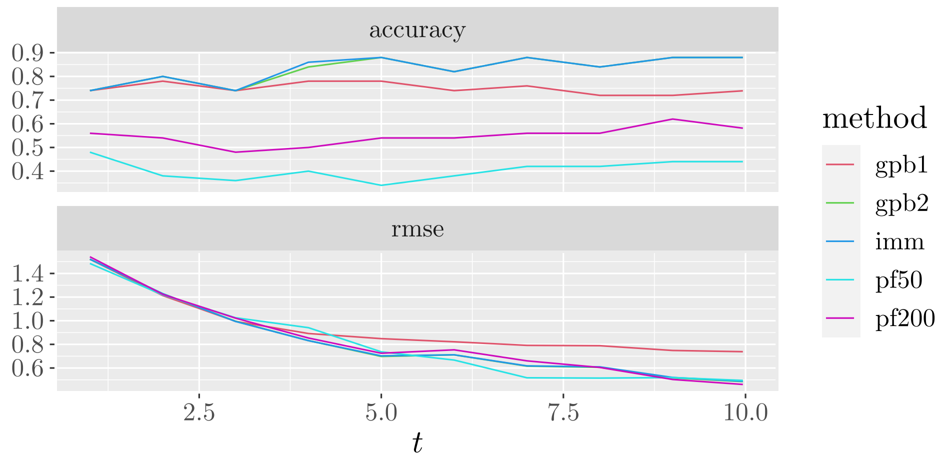
From worst to best on the root mean square error, the GPB1 performs the worst, whereas the remaining approximations perform about the same, and they also overlap with the exact approach. The reason why the particle filter did so well here might be explained by the curse of dimensionality. In this comparison the hidden dynamic variable was a univariate variable, and missing the "correct state" is less likely to be a complete miss compared to if the dynamic variable was multivariate. Whereas missing the correct switching state is in fact a complete miss, and these misses are easier to see when looking at the accuracy.

As GPB2 and IMM performs equally well, and since IMM is more efficient this is the most favorable choice (of the ones we have explored) when dealing with more complex SLDSs.

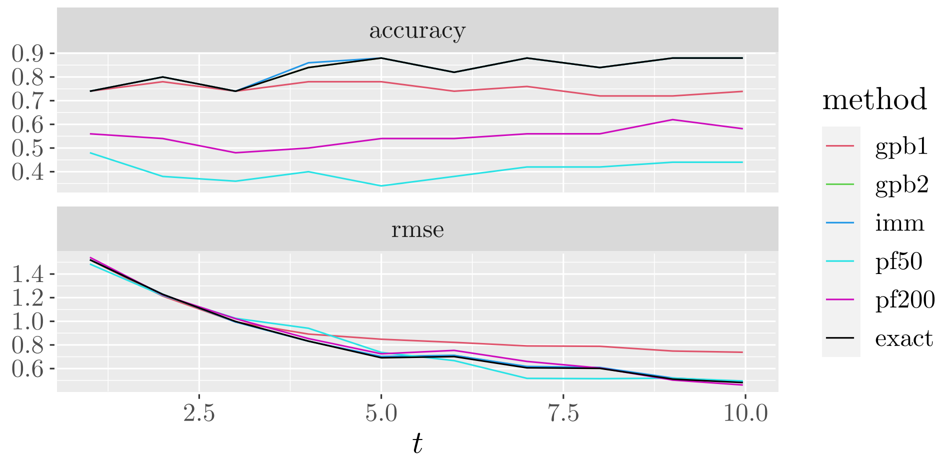
3.5 Linearization of Nonlinear Dynamical Systems

We have until now only dealt with CLGs when we have continuous variables, but unfortunately many real world problems have nonlinear dependencies between the variables. For LDSs this means that we do not end up with any Gaussians, and for SLDSs this means we do not have a mixture of Gaussians any longer.

There are several approaches to deal with the nonlinearity, the EKF is the traditional approach, but there exists other approaches that are more sophisticated and tackles the



(a) Score excluding exact inference.



(b) Score including exact inference.

Figure 3.17: Comparison of the approximations generated from 50 simulations for 10 time steps. The accuracy is computed for the switching state whereas the root mean square error is computed for the dynamic state. For the particle filter we used 50 particles (cyan) and 200 particles (magenta).

problem more directly, such as the *unscented Kalman filter* proposed by Julier and Uhlmann (1997). Both methods are explained in detail by Lerner (2003, pp. 115–130). The unscented Kalman filter appears to outperform the EKF as it is more direct and gives a less biased estimate.

We choose to go with the traditional EKF, as it works well on many nonlinear real world problems, and EKF is a method which I believe is easier to understand than the unscented Kalman filter.

Let $\mathbf{Y}'$ be a random vector of length N in a BN with the parent $\mathbf{Y}$ with the same length, and we assume that $\mathbf{Y}$ is Gaussian with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$. Assume that the CPD of $\mathbf{Y}'$ is $\mathbf{Y}' = \mathbf{f}(\mathbf{Y})$, where $\mathbf{f} : \mathbb{R}^N \to \mathbb{R}^N$ is a nonlinear deterministic function. Assuming that $\mathbf{f}$ is deterministic does not restrict the generality of the method, as we can always add some stochasticity to the function as a source of other random variables, e.g. if

$$\mathbf{Y}' = \begin{bmatrix} \sqrt{Y_1^2 + Y_2^2} + \epsilon_1 \\ \sin Y_1 - \cos Y_2 + \epsilon_2 \end{bmatrix} = \begin{bmatrix} \sqrt{Y_1^2 + Y_2^2} \\ \sin Y_1 - \cos Y_2 \end{bmatrix} + \boldsymbol{\epsilon}, \quad \epsilon \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_{\boldsymbol{\epsilon}}),$$

then we can view $\mathbf{f}$ as a deterministic function with two vectors $\mathbf{f}(\mathbf{Y}, \boldsymbol{\epsilon})$.

Our goal is to find a Gaussian approximation for the joint PDF $p(\mathbf{y}, \mathbf{y}')$. In the EKF, we find a linear approximation $\hat{\mathbf{f}}$ to $\mathbf{f}$. That way, we get a CLG, which we know by Expression (3.15) gives us a joint PDF $p(\mathbf{y}, \mathbf{y}')$ that is Gaussian. We use the standard Taylor expansion around the mean of $\mathbf{Y}$ to approximate $\mathbf{f}(\mathbf{Y}')$:

$$\mathbf{Y}' = \mathbf{f}(\mathbf{Y}) \approx \hat{\mathbf{f}}(\mathbf{Y}) = \mathbf{f}(\boldsymbol{\mu}) + \nabla \mathbf{f}^{\mathsf{T}}(\boldsymbol{\mu})(\mathbf{Y} - \boldsymbol{\mu}), \tag{3.53}$$

where

$$\nabla \mathbf{f}^{\mathsf{T}}(\mathbf{y}) = \begin{bmatrix} \nabla f_1(\mathbf{y}) & \dots & \nabla f_N(\mathbf{y}) \end{bmatrix}^{\mathsf{T}} = \begin{bmatrix} \frac{\partial f_1(\mathbf{y})}{\partial y_1} & \cdots & \frac{\partial f_1(\mathbf{y})}{\partial y_N} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_N(\mathbf{y})}{\partial y_1} & \cdots & \frac{\partial f_N(\mathbf{y})}{\partial y_N} \end{bmatrix}, \tag{3.54}$$

is the Jacobian matrix.

Now we connect this to the KF approach. Instead of $\mathbf{Y}^{(t+1)} \mid \mathbf{Y}^{(t)}$ being a CLG with mean $\mathbf{A}\mathbf{Y}^{(t)} + \mathbf{b}$, it is instead a nonlinear function

$$\mathbf{Y}^{(t+1)} = \mathbf{f}(\mathbf{Y}^{(t)}, \boldsymbol{\epsilon}^{(t+1)}), \quad \boldsymbol{\epsilon}^{(1)}, \boldsymbol{\epsilon}^{(2)}, \ldots \overset{\text{iid}}{\sim} \mathcal{N}(\mathbf{0}, \mathbf{Q}), \tag{3.55}$$

where $\boldsymbol{\epsilon}^{(t)}$ exclusively encodes the stochasticity in the transition, given $\mathbf{Y}^{(t)}$, and where $\boldsymbol{\epsilon}^{(i)}$ and $\mathbf{Y}^{(j)}$ are independent for all $i, j \geq 1$. We want the filtered estimate at each time step to be Gaussian $\mathcal{N}(\boldsymbol{\mu}^{(t)}, \boldsymbol{\Sigma}^{(t)})$. The general filtering procedure is almost the same as the general KF, but with the following modifications:

Assume $\mathbf{Y}^{(t)} \mid \mathbf{o}^{(1:t)}$ is Gaussian, and we want to perform the prediction step to estimate $\mathbf{Y}^{(t+1)} \mid \mathbf{o}^{(1:t)}$. The linear approximation is thus

$$\mathbf{Y}^{(t+1)} \approx \hat{\mathbf{f}}(\mathbf{Y}^{(t)}, \boldsymbol{\epsilon}^{(t+1)}) = \mathbf{f}(\boldsymbol{\mu}^{(t)}) + \nabla \mathbf{f}^{\mathsf{T}}(\boldsymbol{\mu}^{(t)})(\mathbf{Y}^{(t)} - \boldsymbol{\mu}^{(t)}) + \boldsymbol{\epsilon}^{(t+1)},$$

and by the result from Expression (3.16) $\mathbf{Y}^{(t+1)} \mid \mathbf{o}^{(1:t)}$ is approximately Gaussian distributed $\mathcal{N}(\boldsymbol{\mu}^{(\cdot t+1)}, \boldsymbol{\Sigma}^{(\cdot t+1)})$, where

$$\begin{aligned} \boldsymbol{\mu}^{(\cdot t+1)} &= \mathbf{f}(\boldsymbol{\mu}^{(t)}) \\ \boldsymbol{\Sigma}^{(\cdot t+1)} &= \nabla \mathbf{f}^{\mathsf{T}}(\boldsymbol{\mu}^{(t)}) \boldsymbol{\Sigma}^{(t)} \nabla \mathbf{f}(\boldsymbol{\mu}^{(t)}) + \mathbf{Q}. \end{aligned} \tag{3.56}$$

Compare this with the predicted estimate of the regular KF given by Expression (3.20), and we see that $\nabla \mathbf{f}^{\mathsf{T}}(\boldsymbol{\mu}^{(t)})$ has replaced the role of $\mathbf{A}$ and is now time dependent, and

$\mathbf{f}(\boldsymbol{\mu}^{(t)})$ has replaced $\mathbf{A}\boldsymbol{\mu}^{(t)} + \mathbf{b}$. We emphasize that the result can trivially be extended to SLDSs, where we have a nonlinear function $\mathbf{f}_s$ for each switching state $s = 1, \ldots, M$. Assuming the observation model is a CLG, the condition step is unaffected, and we use the result from Expression (3.21) and (3.56). For a nonlinear observation model, we omit the details as this will not be used, but the procedure is similar.

Chapter 4

Application in motion states

In this Chapter we show how to use noisy position data to classify a motion state, such as moving at constant acceleration, constant speed or being stationary (not moving). In Section 4.1 we start by deriving the nonlinear dynamical system for motion in a 2-dimensional plane, where we assume a constant turn rate and acceleration. Later we show how we can add new constraints such as constant speed or being stationary. We follow up with a simulation study where we try to classify whether a car is driving at constant speed, constant acceleration or is stationary, using noisy position data as observations. In Section 4.2 we try to apply the same nonlinear dynamical system, but now with real GPS data taken from a bus trip in Trondheim. Here, the goal is to detect when we are walking and when we are taking the bus, using position data collected from a smartwatch. The smartwatch is from the company Garmin, which is a technology company that specializes in GPS technology for automotive, aviation, marine, outdoor and sport activities (Garmin, 2022a).

4.1 Deriving the dynamical system

From elementary physics, we have the following equations of motion assuming constant acceleration:

$$s(t) = s_0 + v_0 t + \frac{1}{2}at^2 \tag{4.1}$$

$$v(t) = v_0 + at, \tag{4.2}$$

where $s(t)$ is the displacement and $v(t)$ is the velocity at time t. The variable s_0 denotes the initial displacement $s(0)$ and v_0 denotes the initial velocity $v(0)$. Let Δt be a constant, denoting a small time interval, e.g. 1 second. Then from Expression (4.1) and (4.2), we have that

$$\begin{aligned} v(t+\Delta t) &= v_0 + a(t+\Delta t) = v_0 + at + a\Delta t \\ &= v(t) + a\Delta t, \end{aligned} \tag{4.3}$$

and

$$\begin{aligned} s(t+\Delta t) &= s_0 + v_0(t+\Delta t) + \frac{1}{2}a(t+\Delta t)^2 = s_0 + v_0 t + \frac{1}{2}at^2 + (v_0 + at)\Delta t + \frac{1}{2}a\Delta t^2 \\ &= s(t) + v(t)\Delta t + \frac{1}{2}a\Delta t^2, \end{aligned} \tag{4.4}$$

which we can write as a linear system

$$\begin{bmatrix} s(t+\Delta t) \\ v(t+\Delta t) \end{bmatrix} = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} s(t) \\ v(t) \end{bmatrix} + \begin{bmatrix} \frac{1}{2}a\Delta t^2 \\ a\Delta t, \end{bmatrix} \tag{4.5}$$

or

$$\begin{bmatrix} s(t+\Delta t) \\ v(t+\Delta t) \\ a(t+\Delta t) \end{bmatrix} = \begin{bmatrix} 1 & \Delta t & \frac{1}{2}\Delta t^2 \\ 0 & 1 & \Delta t \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} s(t) \\ v(t) \\ a(t) \end{bmatrix}, \tag{4.6}$$

where we have rewritten the constant acceleration as a constant function $a(t + \Delta t) = a(t) = a$. Expression (4.6) will be our basis in the model formulation of the SLDS in this chapter.

Let the dynamic state be given by the vector

$$\mathbf{Y}^{(t)} = \begin{bmatrix} Y^{(t)} \\ V^{(t)} \\ A^{(t)}, \end{bmatrix} \tag{4.7}$$

where the entries correspond to the position, velocity and acceleration of the truck, respectively. We can describe the motion of the truck in one dimension as follows

$$\begin{aligned} \mathbf{y}^{(t+1)} = \begin{bmatrix} y^{(t+1)} \\ v^{(t+1)} \\ a^{(t+1)} \end{bmatrix} &= \begin{bmatrix} y^{(t)} + \Delta t v^{(t)} + \frac{1}{2}\Delta t^2 a^{(t)} \\ v^{(t)} + \Delta t a^{(t)} \\ a^{(t)} + \epsilon_a^{(t)} \end{bmatrix} \\ &= \begin{bmatrix} 1 & \Delta t & \frac{1}{2}\Delta t^2 \\ 0 & 1 & \Delta t \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} y^{(t)} \\ v^{(t)} \\ a^{(t)} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \epsilon_a^{(t)} \end{bmatrix}, \end{aligned} \tag{4.8}$$

where we add some noise $\epsilon_a^{(t)} \sim \mathcal{N}(0, Q_a)$ to the acceleration, to account for the truck not driving exactly at constant velocity. The variable Δt is the time difference between the time steps.

The assumption of a truck driving in a straight direction without any turns is quite narrow. A more useful model include motion in a two-dimensional plane. A simple extension is to let

$$\mathbf{y}^{(t+1)} = \begin{bmatrix} x^{(t+1)} \\ y^{(t+1)} \\ v_x^{(t+1)} \\ v_y^{(t+1)} \\ a_x^{(t+1)} \\ a_y^{(t+1)} \end{bmatrix} = \begin{bmatrix} 1 & 0 & \Delta t & 0 & \frac{1}{2}\Delta t^2 & 0 \\ 0 & 1 & 0 & \Delta t & 0 & \frac{1}{2}\Delta t^2 \\ 0 & 0 & 1 & 0 & \Delta t & 0 \\ 0 & 0 & 0 & 1 & 0 & \Delta t \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x^{(t)} \\ y^{(t)} \\ v_x^{(t)} \\ v_y^{(t)} \\ a_x^{(t)} \\ a_y^{(t)} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ \epsilon_{ax}^{(t)} \\ \epsilon_{ay}^{(t)} \end{bmatrix}. \tag{4.9}$$

This however, causes some problems when we assume the model to have constant acceleration different than zero, and similarly if we assume a less general, constant velocity, model, where we assume the velocity to be constant and different than zero.

Assume we want to model a car of which we expect to drive at constant speed, 90 km/h or 25 m/s. Of course, the car might sometimes drive slightly faster and slightly slower, so we add some noise to the velocity components. In one dimension, we could let

$$v^{(t+1)} = \phi_v v^{(t)} + 25 \cdot (1 - \phi_v) + \epsilon_v^{(t+1)}, \quad \epsilon_v^{(t+1)} \sim \mathcal{N}(0, Q_v),$$

where $0 \leq \phi_v \leq 1$. If $\phi = 0$, we have that the velocities $V^{(1)}, V^{(2)}, \ldots$ are independent with expected value of 25, and if $\phi = 1$, we simply have $V^{(t+1)} \mid V^{(t)} \sim \mathcal{N}(V^{(t)}, Q_v)$, which have the marginal expectation $\mathrm{E}[V^{(t+1)}] = \mathrm{E}[V^{(t)}] = \cdots = \mathrm{E}[V^{(1)}]$, where we choose the initial expectation to be $\mathrm{E}[V^{(1)}] = 25$ by assumption. For $0 < \phi < 1$, we see that we already chose b such that the limiting distribution gives the limiting expectation of 25 m/s, see the result of Expression (3.52). If we extend to two dimensions instead, we have the two velocity components $V_x^{(t)}$ and $V_y^{(t)}$. The roads can go in any direction, so we are forced to let $\mathrm{E}[V_x^{(t)}] = \mathrm{E}[V_y^{(t)}] = 0$. We still want the speed of the car to have an expectation of 25 m/s:

$$\mathrm{E}\left[\sqrt{\left(V_x^{(t)}\right)^2 + \left(V_y^{(t)}\right)^2}\right] = 25. \tag{4.10}$$

Assuming $V_x^{(t)}, V_y^{(t)} \overset{\text{iid}}{\sim} \mathcal{N}(0, \Gamma)$, we need to determine Γ from Expression (4.10), and further we need to determine $\mathrm{Var}[V_x^{(t+1)} \mid V_x^{(t)}]$ and $\mathrm{Var}[V_y^{(t+1)} \mid V_y^{(t)}]$ from Γ. This means that we are forced to set the variance of the velocity components. This restriction will often cause the model to have a much too high variance in the resulting speed ($v = \sqrt{v_x^2 + v_y^2}$), as we increase the expected speed. Therefore, we instead want to have the option to set up a model that encodes the (absolute) velocity without decomposing it into x and y-components, where we are free to adjust the variance ourselves.

The following model is adapted from Kaempchen et al. (2004), where they do an analysis on *Stop-and-Go* situations, which has been extensively studied for adapted cruise control, collision avoidance and collision warning (Vahidi & Eskandarian, 2003). Kaempchen et al. (2004) propose two models, the *free motion model*, where we decompose the velocity and acceleration in x and y-components as we did in Expression (4.9), and the *bicycle model*, which we now introduce.

Let the dynamic state $\mathbf{Y}^{(t)}$ be given by the vector

$$\mathbf{y}^{(t)} = \begin{bmatrix} x^{(t)} \\ y^{(t)} \\ \psi^{(t)} \\ v^{(t)} \\ \omega^{(t)} \\ a^{(t)} \end{bmatrix}, \tag{4.11}$$

where $(x^{(t)}, y^{(t)})$ are UTM-coordinates (Universal Transverse Mercator) in unit meters, $v^{(t)}$, is the speed of the object in unit m/s, $a^{(t)}$ is the acceleration in unit m/s^2, and $\psi^{(t)}$ and $\omega^{(t)}$ are the yaw angle, and yaw rate, respectively. We set the yaw angle to be the the angle between the horizontal axis and the driving direction as shown in Figure 4.1. This naturally leads to a nonlinear dynamical system as we are dealing with angles,

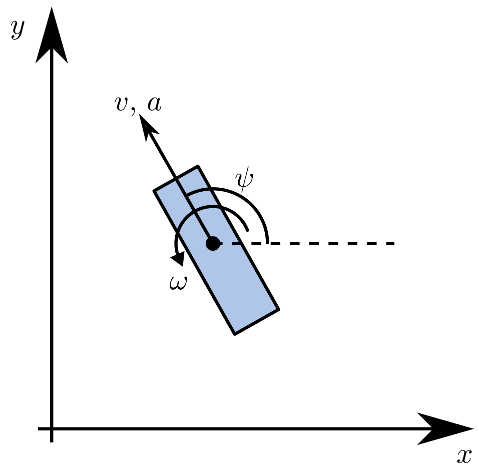


Figure 4.1: Bicycle model showing the kinematic behavior of a vehicle. The blue rectangle represents the vehicle driving in the direction given by the arrow pointing towards v and a. The orientation of the vehicle is determined by the yaw angle ψ and yaw rate ω.

which we solve by the EKF approach described in Section 3.5.

We need to find the nonlinear function $\mathbf{f}$, that describes the dynamical model

$$\mathbf{y}^{(t+1)} = \mathbf{f}(\mathbf{y}^{(t)}, \boldsymbol{\epsilon}^{(t+1)}) = \begin{bmatrix} f_1(\mathbf{y}^{(t)}, \boldsymbol{\epsilon}^{(t+1)}) \\ f_2(\mathbf{y}^{(t)}, \boldsymbol{\epsilon}^{(t+1)}) \\ f_3(\mathbf{y}^{(t)}, \boldsymbol{\epsilon}^{(t+1)}) \\ f_4(\mathbf{y}^{(t)}, \boldsymbol{\epsilon}^{(t+1)}) \\ f_5(\mathbf{y}^{(t)}, \boldsymbol{\epsilon}^{(t+1)}) \\ f_6(\mathbf{y}^{(t)}, \boldsymbol{\epsilon}^{(t+1)}) \end{bmatrix}.$$

From elementary physics, assuming constant acceleration and constant yaw rate, we have

$$v(t) = v_0 + at \tag{4.12}$$
$$\psi(t) = \psi_0 + \omega t \tag{4.13}$$
$$v_x(t) = v(t)\cos(\psi(t)) \tag{4.14}$$
$$v_y(t) = v(t)\sin(\psi(t)), \tag{4.15}$$

where v_0 is the initial velocity, and ψ_0 is the initial yaw angle. We integrate to get an expression for $x(t)$ and $y(t)$:

$$\begin{aligned}
x(t) &= \int v_x(t)\,\mathrm{d}t = \int (v_0 + at)\cos(\psi_0 + \omega t)\,\mathrm{d}t \\
&= \frac{v_0 + at}{\omega}\sin(\psi_0 + \omega t) + \frac{a}{\omega^2}\cos(\psi_0 + \omega t) + C_x; \\
y(t) &= \int v_y(t)\,\mathrm{d}t = \int (v_0 + at)\sin(\psi_0 + \omega t)\,\mathrm{d}t \\
&= -\frac{v_0 + at}{\omega}\cos(\psi_0 + \omega t) + \frac{a}{\omega^2}\sin(\psi_0 + \omega t) + C_y,
\end{aligned}$$

where C_x and C_y are constants, and where we have assumed $\omega \neq 0$. We deal with the case $\omega = 0$ shortly. When we add a small time interval Δt into $x(t)$ and $y(t)$, we get

$$\begin{aligned}
x(t + \Delta t) &= \frac{v_0 + at + a\Delta t}{\omega}\sin(\psi_0 + \omega t + \omega\Delta t) + \frac{a}{\omega^2}\cos(\psi_0 + \omega t + \omega\Delta t) + C_x \\
&= \frac{v(t) + a\Delta t}{\omega}\sin(\psi(t) + \omega\Delta t) + \frac{a}{\omega^2}\cos(\psi(t) + \omega\Delta t) \\
&\quad + \left(x(t) - \left[\frac{v(t)}{\omega}\sin(\psi(t)) + \frac{a}{\omega^2}\cos(\psi(t))\right]\right) \\
&= x(t) + \frac{v(t) + a\Delta t}{\omega}\sin(\psi(t) + \omega\Delta t) - \frac{v(t)}{\omega}\sin(\psi(t)) \\
&\quad + \frac{a}{\omega^2}\left(\cos(\psi(t) + \omega\Delta t) - \cos(\psi(t))\right); \\
y(t + \Delta t) &= -\frac{v_0 + at + a\Delta t}{\omega}\cos(\psi_0 + \omega t + \omega\Delta t) + \frac{a}{\omega^2}\sin(\psi_0 + \omega t + \omega\Delta t) + C_y \\
&= -\frac{v(t) + a\Delta t}{\omega}\cos(\psi(t) + \omega\Delta t) + \frac{a}{\omega^2}\sin(\psi(t) + \omega\Delta t) \\
&\quad + \left(y(t) - \left[-\frac{v(t)}{\omega}\cos(\psi(t)) + \frac{a}{\omega^2}\sin(\psi(t))\right]\right) \\
&= y(t) - \frac{v(t) + a\Delta t}{\omega}\cos(\psi(t) + \omega\Delta t) + \frac{v(t)}{\omega}\cos(\psi(t)) \\
&\quad + \frac{a}{\omega^2}\left(\sin(\psi(t) + \omega\Delta t) - \sin(\psi(t))\right),
\end{aligned}$$

where we used Expression (4.12) and (4.13) in the substitutions. The velocity after a small time interval $v(t + \Delta t)$ is already given by Expression (4.3), and the the result for the yaw angle $\psi(t + \Delta t)$ is similar:

$$\psi(t + \Delta t) = \psi(t) + \omega\Delta t. \tag{4.16}$$

Combining these results, we get the following nonlinear system:

$$\begin{aligned}\mathbf{y}' &= \mathbf{f}(\mathbf{y}, \boldsymbol{\epsilon}') \\ &= \begin{bmatrix} x + \frac{v+a\Delta t}{\omega}\sin(\psi+\omega\Delta t) - \frac{v}{\omega}\sin\psi + \frac{a}{\omega^2}\left(\cos(\psi+\omega\Delta t) - \cos\psi\right) \\ y - \frac{v+a\Delta t}{\omega}\cos(\psi+\omega\Delta t) + \frac{v}{\omega}\cos\psi + \frac{a}{\omega^2}\left(\sin(\psi+\omega\Delta t) - \sin\psi\right) \\ \psi + \omega\Delta t \\ v + a\Delta t \\ \omega + \epsilon'_\omega \\ a + \epsilon'_a \end{bmatrix},\end{aligned} \tag{4.17}$$

where we have added some noise $\epsilon_\omega^{(t)} \sim \mathcal{N}(0, Q_\omega)$ to the yaw rate and to the acceleration $\epsilon_a^{(t)} \sim \mathcal{N}(0, Q_a)$. Assuming no correlation between the noise terms, this is the same as writing

$$\boldsymbol{\epsilon}^{(t)} \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}), \quad \mathbf{Q} = \begin{bmatrix} 0 &&&&& \\ & 0 &&&& \\ && 0 &&& \\ &&& 0 && \\ &&&& Q_\omega & \\ &&&&& Q_a \end{bmatrix} = \operatorname{diag}(0,0,0,0,Q_\omega,Q_a),$$

where diag() is a shorthand notation for a diagonal matrix. The Jacobian is given by

$$\nabla_{\mathbf{y}}\mathbf{f}^{\mathsf{T}}(\mathbf{y}, \boldsymbol{\epsilon}') = \begin{bmatrix} 1 & 0 & -(f_2(\mathbf{y},\boldsymbol{\epsilon}') - y) & \frac{\partial f_1}{\partial v} & \frac{\partial f_1}{\partial \omega} & \frac{\partial f_1}{\partial a} \\ & 1 & f_1(\mathbf{y},\boldsymbol{\epsilon}') - x & \frac{\partial f_2}{\partial v} & \frac{\partial f_2}{\partial \omega} & \frac{\partial f_2}{\partial a} \\ && 1 & 0 & \Delta t & 0 \\ &&& 1 & 0 & \Delta t \\ &&&& 1 & 0 \\ &&&&& 1 \end{bmatrix}, \tag{4.18}$$

where

$$\begin{aligned}\frac{\partial f_1}{\partial v} &= \frac{1}{\omega}(\sin(\psi+\omega\Delta t) - \sin\psi) \\ \frac{\partial f_2}{\partial v} &= -\frac{1}{\omega}(\cos(\psi+\omega\Delta t) - \cos\psi) \\ \frac{\partial f_1}{\partial \omega} &= \frac{v+a\Delta t}{\omega}\Delta t\cos(\psi+\omega\Delta t) - \frac{1}{\omega^2}\left((v+2a\Delta t)\sin(\psi+\omega\Delta t) - v\sin\psi\right) \\ &\quad + \frac{2a}{\omega^3}\left(\cos\psi - \cos(\psi+\omega\Delta t)\right) \\ \frac{\partial f_2}{\partial \omega} &= \frac{v+a\Delta t}{\omega}\Delta t\sin(\psi+\omega\Delta t) + \frac{1}{\omega^2}\left((v+2a\Delta t)\cos(\psi+\omega\Delta t) - v\cos\psi\right) \\ &\quad + \frac{2a}{\omega^3}\left(\sin\psi - \sin(\psi+\omega\Delta t)\right) \\ \frac{\partial f_1}{\partial a} &= \frac{\Delta t}{\omega}\sin(\psi+\omega\Delta t) + \frac{1}{\omega^2}\left(\cos(\psi+\omega\Delta t) - \cos\psi\right) \\ \frac{\partial f_2}{\partial a} &= -\frac{\Delta t}{\omega}\cos(\psi+\omega\Delta t) + \frac{1}{\omega^2}\left(\sin(\psi+\omega\Delta t) - \sin\psi\right),\end{aligned}$$

given that $\omega \neq 0$. We now have to deal with the case for when $\omega \approx 0$. We solve this by Taylor expanding $f_1(\mathbf{y}, \boldsymbol{\epsilon}')$ and $f_2(\mathbf{y}, \boldsymbol{\epsilon}')$ about $\omega = 0$, giving

$$f_1(\mathbf{y}, \boldsymbol{\epsilon}') = x + \frac{1}{2}\Delta t(2v + a\Delta t)\cos\psi - \frac{1}{6}\omega\Delta t^2(3v + 2a\Delta t)\sin\psi + \mathcal{O}(\omega^2); \tag{4.19}$$
$$f_2(\mathbf{y}, \boldsymbol{\epsilon}') = y + \frac{1}{2}\Delta t(2v + a\Delta t)\sin\psi + \frac{1}{6}\omega\Delta t^2(3v + 2a\Delta t)\cos\psi + \mathcal{O}(\omega^2). \tag{4.20}$$

After some experimentation with numerical accuracy, when $|\omega| < 10^{-4}$, it appears to better to swap to the Taylor series approximation for $f_1(\mathbf{y}, \boldsymbol{\epsilon}')$ and $f_2(\mathbf{y}, \boldsymbol{\epsilon}')$. For $|\omega| < 10^{-4}$, the Jacobian has the same form given by Expression (4.18), but now with

$$\begin{aligned}
\frac{\partial f_1}{\partial v} &= \Delta t\cos\psi - \frac{1}{2}\omega\Delta t^2\sin\psi\\
\frac{\partial f_2}{\partial v} &= \Delta t\sin\psi + \frac{1}{2}\omega\Delta t^2\cos\psi\\
\frac{\partial f_1}{\partial \omega} &= -\frac{1}{6}(3v + 2a\Delta t)\Delta t^2\sin\psi - \frac{1}{12}\omega(4v + 3a\Delta t)\Delta t^3\cos\psi\\
\frac{\partial f_2}{\partial \omega} &= \frac{1}{6}(3v + 2a\Delta t)\Delta t^2\cos\psi - \frac{1}{12}\omega(4v + 3a\Delta t)\Delta t^3\sin\psi\\
\frac{\partial f_1}{\partial a} &= \frac{1}{2}\Delta t^2\cos\psi - \frac{1}{3}\omega\Delta t^3\sin\psi\\
\frac{\partial f_2}{\partial a} &= \frac{1}{2}\Delta t^2\sin\psi + \frac{1}{3}\omega\Delta t^3\cos\psi.
\end{aligned} \tag{4.21}$$

To summarize, we have for constant acceleration and constant yaw rate:

$$\begin{gathered}
\mathbf{y} = \begin{bmatrix} x\\ y\\ \psi\\ v\\ \omega\\ a \end{bmatrix}, \quad \boldsymbol{\epsilon}' = \begin{bmatrix} 0\\ 0\\ 0\\ 0\\ \epsilon'_\omega\\ \epsilon'_a \end{bmatrix}, \quad \mathrm{Var}[\boldsymbol{\epsilon}'] = \mathbf{Q} = \mathrm{diag}(0,0,0,0,Q_\omega,Q_a),\\
\mathbf{f}(\mathbf{y}, \boldsymbol{\epsilon}') = \begin{bmatrix} f_1(\mathbf{y}, \boldsymbol{\epsilon}')\\ f_2(\mathbf{y}, \boldsymbol{\epsilon}')\\ f_3(\mathbf{y}, \boldsymbol{\epsilon}')\\ f_4(\mathbf{y}, \boldsymbol{\epsilon}')\\ f_5(\mathbf{y}, \boldsymbol{\epsilon}')\\ f_6(\mathbf{y}, \boldsymbol{\epsilon}') \end{bmatrix}, \quad \nabla_{\mathbf{y}}\mathbf{f}^{\mathsf{T}}(\mathbf{y}, \boldsymbol{\epsilon}') = \begin{bmatrix} 1 & 0 & \frac{\partial f_1}{\partial \psi} & \frac{\partial f_1}{\partial v} & \frac{\partial f_1}{\partial \omega} & \frac{\partial f_1}{\partial a}\\ & 1 & \frac{\partial f_2}{\partial \psi} & \frac{\partial f_2}{\partial v} & \frac{\partial f_2}{\partial \omega} & \frac{\partial f_2}{\partial a}\\ & & 1 & 0 & \Delta t & 0\\ & & & 1 & 0 & \Delta t\\ & & & & 1 & 0\\ & & & & & 1 \end{bmatrix},
\end{gathered} \tag{4.22}$$

where the partial derivatives are given by Expression (4.19) or (4.21) for $\omega = 0$. For higher constraints, such as constant velocity, we set all the entries for acceleration equal to zero, and add some noise to the velocity instead as follows:

$$\mathbf{y} = \begin{bmatrix} x \\ y \\ \psi \\ v \\ \omega \\ 0 \end{bmatrix}, \quad \boldsymbol{\epsilon}' = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \epsilon'_v \\ \epsilon'_\omega \\ 0 \end{bmatrix}, \quad \mathrm{Var}[\boldsymbol{\epsilon}'] = \mathbf{Q} = \mathrm{diag}(0,0,0,Q_v,Q_\omega,0),$$

$$\mathbf{f}(\mathbf{y},\boldsymbol{\epsilon}') = \begin{bmatrix} f_1(\mathbf{y},\boldsymbol{\epsilon}') \\ f_2(\mathbf{y},\boldsymbol{\epsilon}') \\ f_3(\mathbf{y},\boldsymbol{\epsilon}') \\ f_4(\mathbf{y},\boldsymbol{\epsilon}') \\ f_5(\mathbf{y},\boldsymbol{\epsilon}') \\ 0 \end{bmatrix}, \quad \nabla_{\mathbf{y}} \mathbf{f}^{\mathsf{T}}(\mathbf{y},\boldsymbol{\epsilon}') = \begin{bmatrix} 1 & 0 & \frac{\partial f_1}{\partial \psi} & \frac{\partial f_1}{\partial v} & \frac{\partial f_1}{\partial \omega} & 0 \\ & 1 & \frac{\partial f_2}{\partial \psi} & \frac{\partial f_2}{\partial v} & \frac{\partial f_2}{\partial \omega} & 0 \\ & & 1 & 0 & \Delta t & 0 \\ & & & 1 & 0 & 0 \\ & & & & 1 & 0 \\ & & & & & 0 \end{bmatrix}, \tag{4.23}$$

For even higher constraints, such as being stationary, we additionally set the velocity and the yaw rate components to zero, and add noise to the x and y-positions and the the yaw angle:

$$\mathbf{y} = \begin{bmatrix} x \\ y \\ \psi \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad \boldsymbol{\epsilon}' = \begin{bmatrix} \epsilon'_x \\ \epsilon'_y \\ \epsilon'_\psi \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad \mathrm{Var}[\boldsymbol{\epsilon}'] = \mathbf{Q} = \mathrm{diag}(Q_x,Q_y,Q_\psi,0,0,0),$$

$$\mathbf{f}(\mathbf{y},\boldsymbol{\epsilon}') = \begin{bmatrix} f_1(\mathbf{y},\boldsymbol{\epsilon}') \\ f_2(\mathbf{y},\boldsymbol{\epsilon}') \\ f_3(\mathbf{y},\boldsymbol{\epsilon}') \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad \nabla_{\mathbf{y}} \mathbf{f}^{\mathsf{T}}(\mathbf{y},\boldsymbol{\epsilon}') = \begin{bmatrix} 1 & 0 & \frac{\partial f_1}{\partial \psi} & 0 & 0 & 0 \\ & 1 & \frac{\partial f_2}{\partial \psi} & 0 & 0 & 0 \\ & & 1 & 0 & 0 & 0 \\ & & & 0 & 0 & 0 \\ & & & & 0 & 0 \\ & & & & & 0 \end{bmatrix}. \tag{4.24}$$

In the following simulation study, we try to determine whether a car is stationary, driving at constant speed, or accelerating, and also we want to estimate its true position given noisy position data. Let $S^{(t)} \in \{1,2,3\}$ corresponding to the states *constant acceleration*, *constant velocity* and *stationary*, respectively. Let $\mathbf{Y}^{(t)}$ be the dynamic variable following the transition model given by Expression (4.22), (4.23) and (4.24), for the states 1,2 and 3, respectively and let

$$\mathbf{O}^{(t)} = \begin{bmatrix} O_x^{(t)} \\ O_y^{(t)} \end{bmatrix}, \tag{4.25}$$

where $O_x^{(t)}$ and $O_y^{(t)}$ are observed x and y-coordinates in unit meters.

The model parameters for the initial distribution are

$$P(S^{(1)}) = \boldsymbol{\pi} = \begin{bmatrix}1/3 & 1/3 & 1/3\end{bmatrix}^\mathsf{T}, \tag{4.26}$$

$$\mathbf{Y}^{(1)} \mid \{S^{(1)} = s\} \sim \mathcal{N}(\boldsymbol{\nu}_s, \boldsymbol{\Gamma}_s), \tag{4.27}$$

$$\boldsymbol{\nu}_1 = \boldsymbol{\nu}_2 = \begin{bmatrix}0 & 0 & 0 & 25/3.6 & 0 & 0\end{bmatrix}^\mathsf{T}, \quad \boldsymbol{\nu}_3 = \mathbf{0}, \tag{4.28}$$

$$\begin{aligned}\boldsymbol{\Gamma}_1 &= \mathrm{diag}(100, 100, 5, 20/3.6, 0.001, 2)^2,\\ \boldsymbol{\Gamma}_2 &= \mathrm{diag}(100, 100, 5, 20/3.6, 0.001, 0)^2,\\ \boldsymbol{\Gamma}_3 &= \mathrm{diag}(100, 100, 5, 0, 0, 0)^2.\end{aligned} \tag{4.29}$$

- In Expression (4.26) we assume that the car can be in any state in the starting time.

- In Expression (4.28), we assume that given that the car is stationary ($S^{(1)} = 3$), we expect it to be at position $(x, y) = (0, 0)$, with 0 yaw angle, velocity, yaw rate and acceleration, whereas when the car is moving, either with constant velocity or constant acceleration, we expect a velocity of 25 km/h, i.e. $\mathrm{E}[V_s^{(1)}] = 25/3.6$ m/s, for $s = 1, 2$. As we cannot tell whether the car is increasing in speed or decreasing, we must set $\mathrm{E}[A_1^{(1)}] = 0$ m/s^2.

- In Expression (4.29), we assume that the x and y-coordinates lie between -200 and 200 meters with 0.95 probability, i.e. about two standard deviations, giving $\mathrm{Var}[X_s^{(t)}] = \mathrm{Var}[Y_s^{(1)}] = 100^2$ for all states. The yaw angle should preferably be uniform in the range $[-\pi, \pi]$, but this is not possible assuming the yaw angle is Gaussian, so we set $\mathrm{Var}[\psi_s] = 5^2$ for all states. For the stationary state, everything else has zero variance. We assume the velocity to be in the range 25 plus-or-minus 40 km/h with .05 probability by setting $\mathrm{Var}[V_s^{(1)}] = (20/3.6)^2$. This, unfortunately may cause a negative velocity during the simulation, so here we must do some "cherry picking" when we select a simulation we are happy with. We assume the car is turning not too rapidly, so we set the variance of the yaw rate very low $\mathrm{Var}[\omega_s^{(1)}] = 0.001^2$, for $s = 1, 2$. Finally, only for constant acceleration, we assume it may lie in the range $[-4, 4]$ m/s^2 with 0.95 probability, i.e. $\mathrm{Var}[A_1^{(1)}] = 2^2$ m/s^2.

The parameters for the transition model are

$$P(S' \mid S) = \mathbf{P} = \begin{bmatrix} 1 - \frac{\Delta t}{\lambda_{\mathrm{CA}}} & \frac{1}{2}\frac{\Delta t}{\lambda_{\mathrm{CA}}} & \frac{1}{2}\frac{\Delta t}{\lambda_{\mathrm{CA}}} \\ \frac{\Delta t}{\lambda_{\mathrm{CV}}} & 1 - \frac{\Delta t}{\lambda_{\mathrm{CV}}} & 0 \\ \frac{\Delta t}{\lambda_{\mathrm{ST}}} & 0 & 1 - \frac{\Delta t}{\lambda_{\mathrm{ST}}} \end{bmatrix} = \begin{bmatrix} 0.95 & 0.025 & 0.025 \\ 0.0125 & 0.9875 & 0 \\ 0.0167 & 0 & 0.983 \end{bmatrix}, \tag{4.30}$$

$$\begin{aligned}\mathbf{Q}_1 &= \mathrm{diag}(0, 0, 0, 0, 0.005, 1)^2\\ \mathbf{Q}_2 &= \mathrm{diag}(0, 0, 0, 0.05, 0.005, 0)^2\\ \mathbf{Q}_3 &= \mathrm{diag}(0, 0, 0, 0, 0, 0)^2\end{aligned} \tag{4.31}$$

- In the transition model for the switching state (see Figure 4.2) given by Expression (4.30) we have made the following assumptions of expected time being in some state:

$$\begin{aligned} \lambda_{\text{CA}} &= 10 \quad \text{mean time (seconds) spent with constant acceleration} \\ \lambda_{\text{CV}} &= 40 \quad \text{mean time (seconds) spent in with constant velocity} \\ \lambda_{\text{ST}} &= 30 \quad \text{mean time (seconds) spent stationary,} \end{aligned} \tag{4.32}$$

  and where we have set $\Delta t = 0.5$ seconds. As the transition model follows the Markov assumption, we can look at each state separately, and view each time step as an independent trial, which leads to a geometric distribution. The expected number of trials before a "success" occurs, i.e. leaving the state, is the reciprocal of the "success" probability. In our case, the expected number of trials is the same as the ratio of the expected time being in state s and the length of the time step, $\lambda_s/\Delta t$, and thus the probability of leaving state s is $\Delta t/\lambda_s$. Furthermore, we assume that the car can not transition directly between the states 2 and 3. That is, the car can not all of a sudden have a constant velocity (different than zero) after being stationary and vice versa, without accelerating first. We have also assumed an equal chance of transitioning to being stationary or having constant velocity, given that the current state is constant acceleration.

- In Expression (4.31) we have made the noise very small, and for the stationary stationary state, the transition is completely deterministic with zero variance. The values were chosen by trial and error to get a good simulation run.

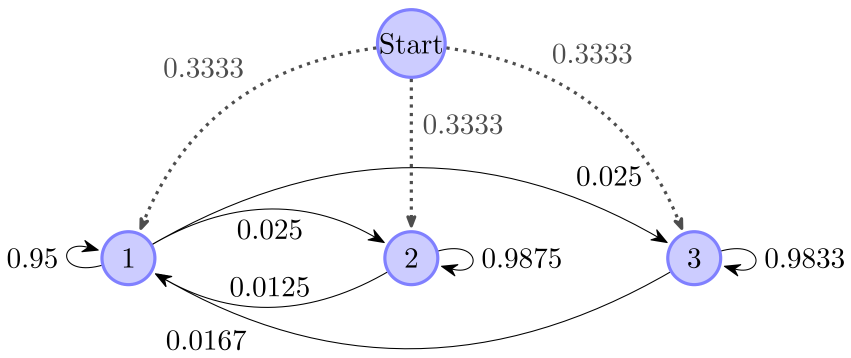


Figure 4.2: Possible initial states $P(S^{(1)})$ and state transitions, $P(S' \mid S)$. The switching states are 1, 2 and 3, corresponding to *constant acceleration*, *constant velocity* and *stationary*, respectively.

Finally, the observation model is given as

$$\begin{aligned} \mathbf{O}^{(t)} \mid \mathbf{Y}^{(t)} &\sim \mathcal{N}(\mathbf{C}\mathbf{Y}^{(t)}, \mathbf{R}) \\ \mathbf{C} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}&, \quad \mathbf{R} = \operatorname{diag}(1.5, 1.5)^2, \end{aligned} \tag{4.33}$$

such that $\mathbf{CY}^{(t)}$ gives the x and y-coordinates, and where we assume the observed position to lie within 3 meters in both x and y-axes with 0.95 probability.

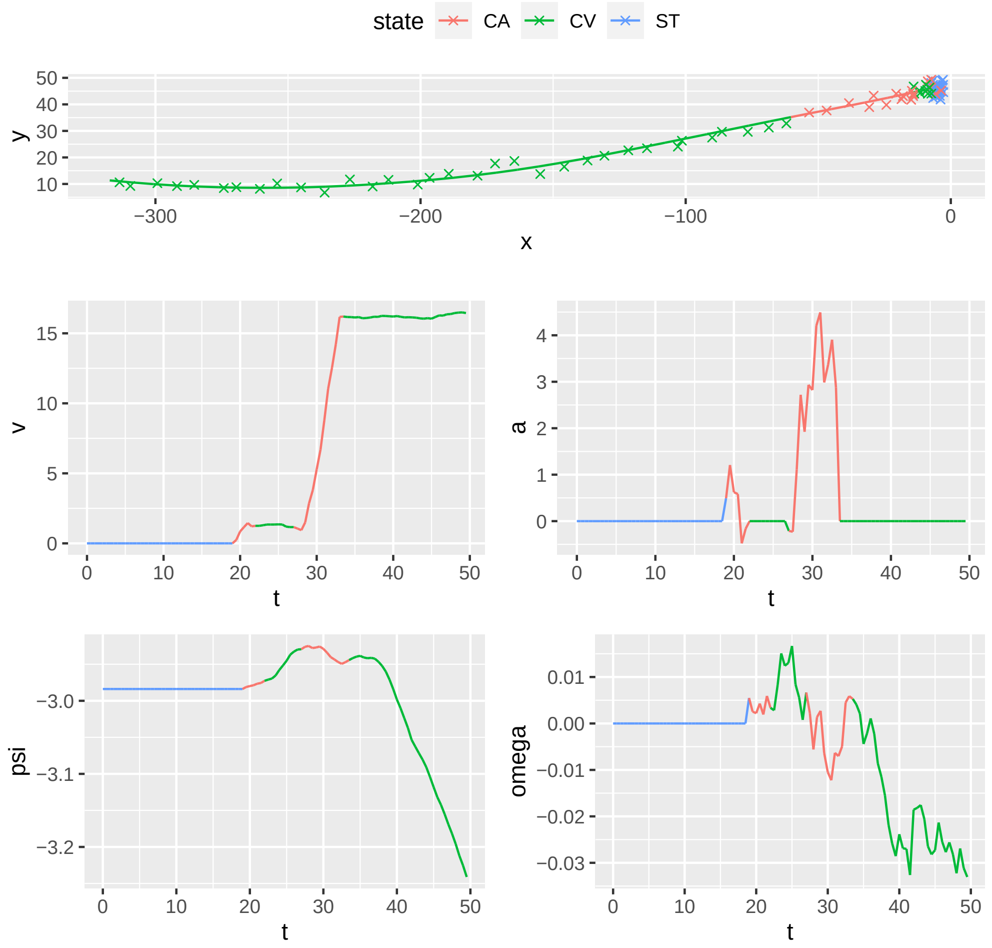
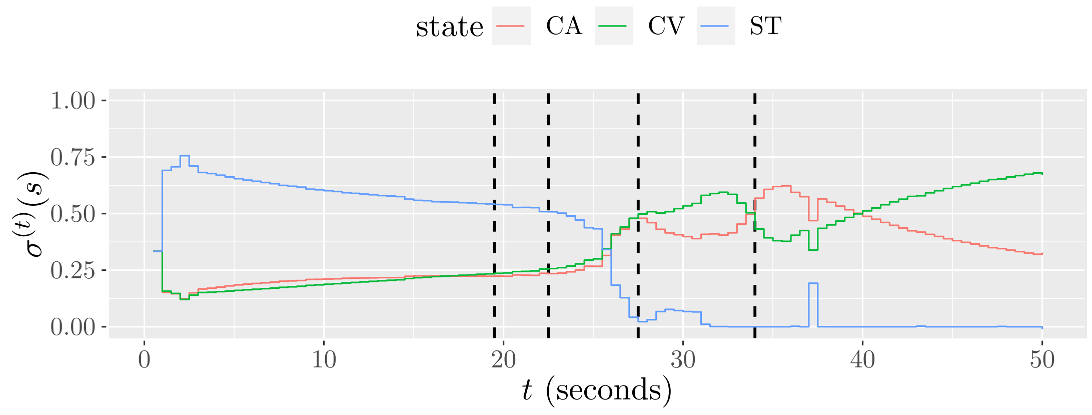
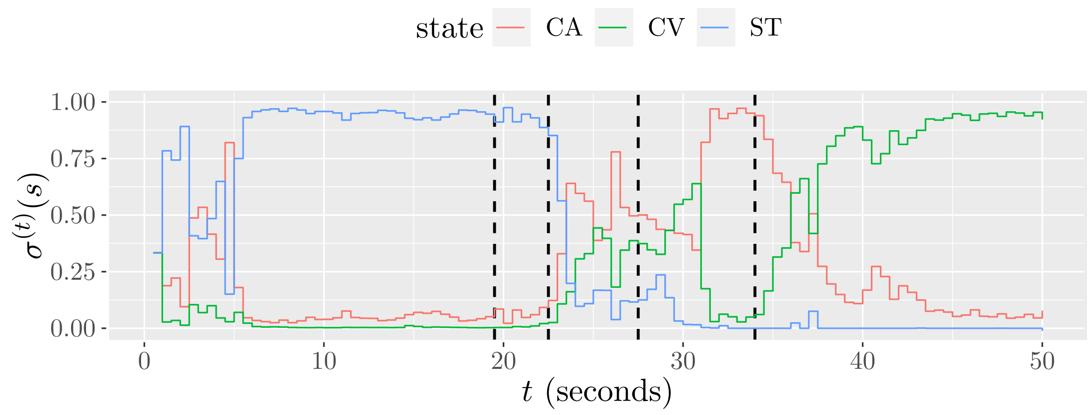


Figure 4.3: Simulation of planar motion, the top plot shows the true trajectory of the car (solid line), and the observed positions are given as crossed points. The velocity, acceleration, yaw angle and yaw rate are shown in the plots in the upper left, upper right, lower left and lower right, respectively. The colors indicate the actual switching state of the car.

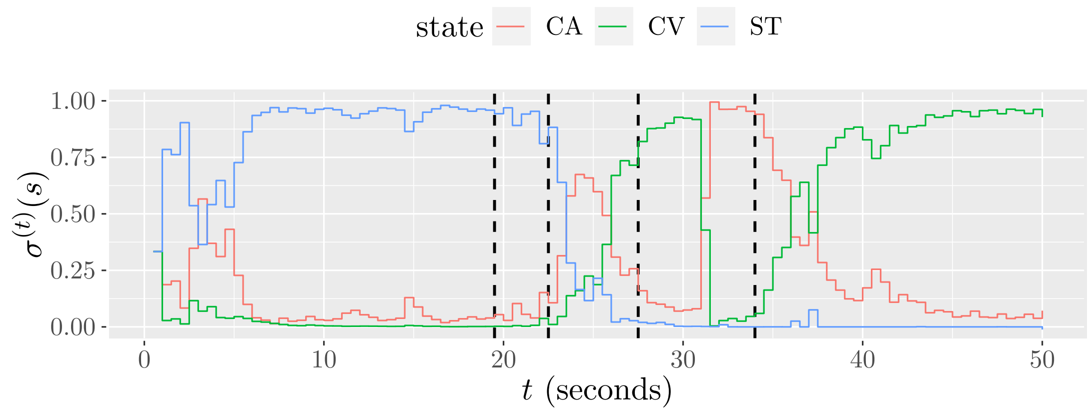
Figure 4.3 shows a simulation run for 50 seconds, or 100 time steps as we choose $\Delta t = 0.5$. The car starts in the upper right corner in the stationary state before it starts moving towards the lower left in the following sequence: stationary$\rightarrow$constant acceleration$\rightarrow$constant velocity$\rightarrow$constant acceleration$\rightarrow$constant velocity, where the car reaches a velocity of about 16 m/s or 60 km/h. The initial yaw angle is around -3 radians which is the same as going directly west, then it is turned anticlockwise (increases) a tiny



(a) GPB1.



(b) GPB2.



(c) IMM.

Figure 4.4: Filtered estimate of the switching state using three different approximations. The vertical dashed lines denote when the actual switch occur.

bit, which is unnoticeable by looking at the trajectory. Then the angle goes clockwise, meaning that the car turns right. The yaw rate can be thought of as the orientation of the steering wheel, where a positive rate means that the steering wheel is turned anticlockwise, and negative rate means that the wheel is turned clockwise.

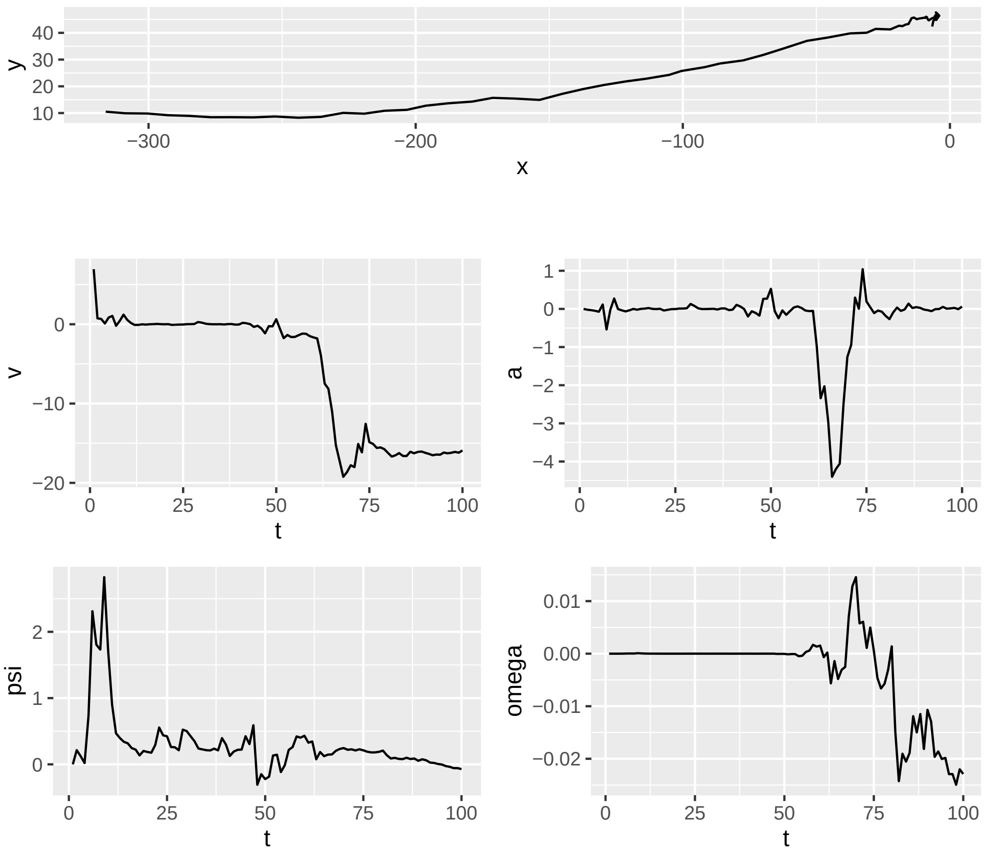


Figure 4.5: Filtered estimate of the dynamic state where IMM is used.

We run the filtering algorithm for the three approximations GPB1, GPB2 and IMM. We omit the particle filter as it underperformed, even with 200 particles for a univariate case in Section 3.4.5. The filtered estimate of the switching state is shown in Figure 4.4 for the three approximations. First of all, GPB1 stands out as being very uncertain on which state it the car is in at all times. GPB2 and IMM looks very similar where the only noticeable difference is in the fourth switching state, when the car accelerates. GPB2 performs slightly better in this case, as it is more uncertain if the car is driving at constant speed or constant acceleration where the IMM is (wrongfully) more certain that the car is driving at constant speed in the start. All three misses to notice the first switch, when the car starts to accelerate for a couple of seconds. This is not very surprising if we look at the trajectory in Figure 4.3. Although the car is accelerating, we do not notice any change in the observation points (orange color). We first notice that the observation points are moving when the car is driving at constant speed (green color). This is reflected in GPB2 and IMM, and partly in GPB1. However, they wrongfully predict that the car is in the constant acceleration state. This is also not surprising if we remember the possible transitions between the states shown in Figure 4.2. A stationary car can not drive at a constant speed (different from zero) without accelerating first. In the last switching state, all approximations correctly predicts the the constant velocity state, where the probability grows more rapidly for GPB2 and IMM then for GPB1.

Figure 4.5 shows the filtered dynamic state for the IMM approximation. The trajectory looks rather smooth compared to the observation points, and it similar to the true trajectory. An interesting observation is that the velocity turns out to be negative, and the angle appears to be around zero radians, so the model believes that the car driving backwards. We have not specified in the model that the car should have a positive velocity, so whether the velocity is positive is "random", or rather is likely to depend on the initial yaw angle, as we assumed that the initial yaw angle followed a normal distribution with mean at zero radians. However, it does not matter whether the model believes the car is driving forwards or backwards, when we are interested in the switching state, or the trajectory.

4.2 Bus and Walk Example

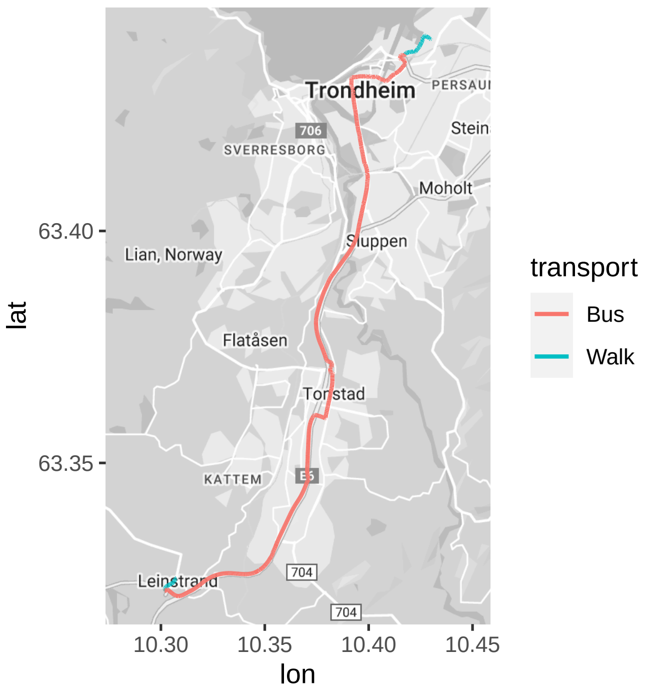
Now we attempt a slightly more complicated example, where we use real data. Here, the goal is to detect whether we are walking or taking the bus, from position data.

The data we use are GPS coordinates received from a Garmin smartwatch, when traveling from Leinstrand to Lademoen in Trondheim. The map view is shown in Figure 4.6. The trip starts by walking towards the bus stop indicated by the blue trajectory in Figure 4.6b. Then after waiting a few minutes, the bus is boarded and it drives to Lademoen before the bus is exited, shown in Figure 4.6c.

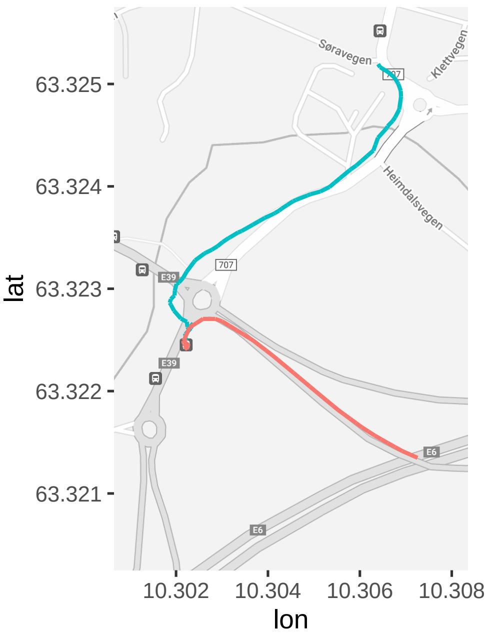
The first task is to actually retrieve this data. When the smartwatch is connected to the Internet, the GPS data is uploaded to Garmin's website where it can be downloaded. However, they do not provide a CSV (Comma-Separated Values)-file format. They instead provide GPX (GPS Exchange), KML (Keyhole Markup Language) and FIT (Flexible and Interoperable Data Transfer), where the latter is their own file format. Garmin provide a software that converts from FIT to CSV (Garmin, 2022b). Using this software, we obtain the position data in CSV-format. The position data use semicircles as a unit. Semicircles are represented by a 32-bit number (Microsoft Docs, 2022), which we convert to degrees using the following formula

$$\text{degrees} = \text{semicircles} \cdot \frac{180}{2^{31}}. \tag{4.34}$$

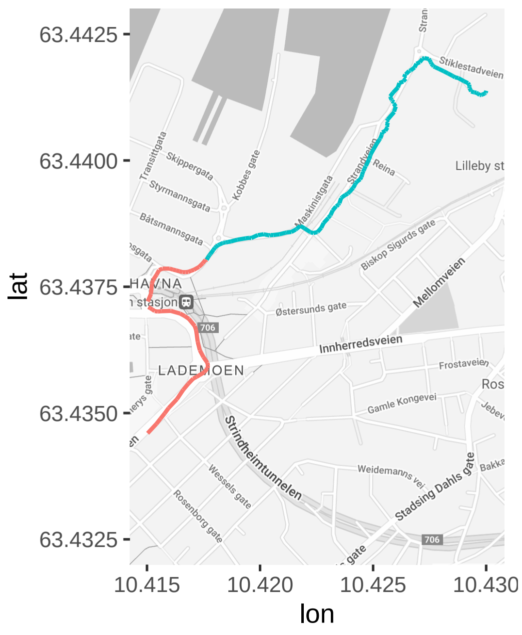
We use the R package `sf` (Pebesma, 2018), to convert the data to a spatial object where we register the coordinate reference system to be EPSG:4326 which is the World



(a) Full trip.



(b) Start trip.



(c) End trip.

Figure 4.6: Map showing the trip starting at Leinstrand and ends in Trondheim at Lademoen, where the colors indicate the transport type, bus (red) and walking (blue). The maps are obtained using the R library `ggmap` (Kahle & Wickham, 2013).

Geodetic System 1984 (WGS84) used in GPS. We then transform to a more suitable coordinate reference system, EPSG:25832, a projection to UTM coordinates in zone 32, which is the recommended projection (Geonorge, 2022), for the southern part of Norway until the county Trøndelag, of which Trondheim lies in. This projection uses the unit meters, which is exactly what we need for the model.

The data contains 1320 observations for a period of fifty minutes. The time interval between each observation is an integer in seconds, the shortest and the most frequent time interval is one second, and the longest time interval is at 12 seconds. This means that we set the time interval to $\Delta t = 1$ second. When there is a time interval higher than this, we add rows of "missing data", such that the filtering algorithm only does the prediction step and skips the condition step when missing data is encountered. This gives us 3000 observations of which 3000-1320=1680 are missing.

The data is ready and we can set up the model. Let $S^{(t)} \in \{\mathrm{WS}, \mathrm{WM}, \mathrm{BS}, \mathrm{BM}\}$, which correspond to the states walk-stationary, walk-moving, bus-stationary and bus-moving, respectively. The first two states represent walking WM, and waiting for the bus WS, respectively, and similarly, BM is the state when we are in a moving bus, and BS is the state when the bus is not moving, usually at a bus stop. The dynamic state $\mathbf{Y}^{(t)}$ uses the nonlinear bicycle form given by Expression (4.11). When being stationary (WS or BS), we use the stationary model dynamics given by Expression (4.24), and when the bus is moving (BM), we use the most general model dynamics where we assume constant acceleration and constant yaw rate, which is given by Expression (4.22). When walking (WM), we assume constant velocity model dynamics given by Expression (4.23), with the extra assumption that the yaw angle is "constant", where we add some noise $\epsilon_\psi \sim \mathcal{N}(0, Q_\psi)$, and let all the entries for the yaw rate be zero.

The model parameters for the initial distribution are

$$P(S^{(1)}) = \boldsymbol{\pi} = \begin{bmatrix} 0.01 & 0.97 & 0.01 & 0.01 \end{bmatrix}^\intercal, \tag{4.35}$$

$$\mathbf{Y}^{(1)} \mid \{S^{(1)} = s\} \sim \mathcal{N}(\boldsymbol{\nu}_s, \boldsymbol{\Gamma}_s), \tag{4.36}$$

$$\begin{aligned} \boldsymbol{\nu}_1 &= \begin{bmatrix} 568273 & 7028843 & 0 & 0 & 0 & 0 \end{bmatrix}^\intercal, \\ \boldsymbol{\nu}_2 &= \begin{bmatrix} 568273 & 7028843 & 0 & 1.5 & 0 & 0 \end{bmatrix}^\intercal, \\ \boldsymbol{\nu}_3 &= \begin{bmatrix} 568273 & 7028843 & 0 & 0 & 0 & 0 \end{bmatrix}^\intercal, \\ \boldsymbol{\nu}_4 &= \begin{bmatrix} 568273 & 7028843 & 0 & 20 & 0 & 0 \end{bmatrix}^\intercal, \end{aligned} \tag{4.37}$$

$$\begin{aligned} \boldsymbol{\Gamma}_1 &= \mathrm{diag}(5000, 5000, \pi, 0, 0, 0)^2, \\ \boldsymbol{\Gamma}_2 &= \mathrm{diag}(5000, 5000, \pi, 1, 0, 0)^2, \\ \boldsymbol{\Gamma}_3 &= \mathrm{diag}(5000, 5000, \pi, 0, 0, 0)^2, \\ \boldsymbol{\Gamma}_4 &= \mathrm{diag}(5000, 5000, \pi, 10, 0.5, 0.5)^2. \end{aligned} \tag{4.38}$$

- In Expression (4.35) we assume that there is a very high probability we start by walking (and moving).
- In Expression (4.37), we expect to be somewhere between Leinstrand and Trond-

heim, so we set the expected position to be at the UTM 32 coordinate (568273, 7028843) which is somewhere in the middle of the map in Figure 4.6a. The expected yaw angle, acceleration and yaw rate is set to zero for the same reasons as described in Section 4.1. When being stationary (either bus or walk), we have an expected velocity of zero. The average walking pace is about 1.5 m/s, and we expect the bus to have an average velocity of 20 m/s when it is moving.

- In Expression (4.38), due to the large travel distance, we added a large variance for the position, where we assume we are 10 km within the expected x and y coordinate with 0.95 probability. We also added a somewhat arbitrary variance to the yaw angle for each state to π^2. When walking (and moving), we expect the walking pace to be in the range $[-0.5, 3.5]$ m/s with 0.95 probability, where again have a possibility of negative velocity. As we saw in Section 4.1, negative velocities does not pose a problem for us. When the bus is moving, we expect the velocity to be in the range $[0, 40]$ m/s with 0.95, probability. Finally, we add some uncertainty to the acceleration and yaw rate for the state BM.

These parameters are gross intuitive guesses. As this is the initial distribution and we have a lot of observation data to guide us at later time steps, it is not crucial to have reasonable inputs.

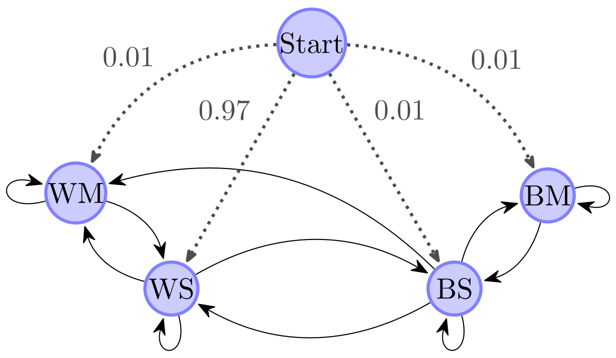


Figure 4.7: Possible initial states $P(S^{(1)})$ and state transitions, $P(S' \mid S)$. The switching states are WS, WM, BS and BM, corresponding to walk-stationary, walk-moving, bus-stationary and bus-moving, respectively. The transition probabilities are shown in Expression (4.30).

The parameters for the transition model are

$$P(S' \mid S) = \mathbf{P} = \begin{bmatrix} 0.99667 & 0.00007 & 0.00327 & 0 \\ 0.00167 & 0.99833 & 0 & 0 \\ 0.00067 & 0.00600 & 0.93333 & 0.06000 \\ 0 & 0 & 0.00833 & 0.99167 \end{bmatrix}, \tag{4.39}$$

$$\begin{aligned} \mathbf{Q}_1 &= \mathrm{diag}(0.15, 0.15, 0.01, 0, 0, 0)^2, \\ \mathbf{Q}_2 &= \mathrm{diag}(0, 0, 0.15, 0.10, 0, 0)^2, \\ \mathbf{Q}_3 &= \mathrm{diag}(0.10, 0.10, 0, 0, 0, 0)^2, \\ \mathbf{Q}_4 &= \mathrm{diag}(0, 0, 0, 0, 0.05, 2.5)^2, \end{aligned} \tag{4.40}$$

- The possible transitions for the switching state are shown in Figure 4.7 and the transition probabilities are given by Expression (4.30). We have assumed that before we enter the bus, we are arriving at the bus stop a short while before the bus arrives, so a direct transition WM→BS is not possible. Also, we cannot enter a moving bus, so the transitions WM→BM and WS→BM are not possible. The same goes for the other way, we cannot exit a moving bus so the transitions BM→WM and BM→WS are not possible. All other transitions are possible, where we have set the probabilities by "guesstimating" the mean time expected in each state as follows

$$\begin{aligned} \lambda_{\mathrm{WS}} &= 5 \cdot 60 && \text{mean time (seconds) spent in WS} \\ \lambda_{\mathrm{WM}} &= 10 \cdot 60 && \text{mean time (seconds) spent in WM} \\ \lambda_{\mathrm{BS}} &= 15 && \text{mean time (seconds) spent in BS} \\ \lambda_{\mathrm{BM}} &= 2 \cdot 60 && \text{mean time (seconds) spent in BM.} \end{aligned} \tag{4.41}$$

  Given that we left the WS state, we assume there is a 1/50 probability of walking again (WM), and otherwise a 49/50 probability of entering the bus (BS). The computation is done as follows

$$P(S^{(t+1)} = \mathrm{WM} \mid S^{(t)} = \mathrm{WS}) = \frac{\Delta t}{\lambda_{\mathrm{WS}}} \cdot \frac{1}{50} = 0.00007.$$

  Also, given that we left the BS state, we assume there is a 1/100 probability of leaving the bus and not move (WS), as we might wait for another bus, a 9/100 probability of leaving the bus and start walking (WM) and a 9/10 probability of staying on the bus when it starts moving again (BM).

- The parameters in Expression (4.40) are the trickiest part to determine, and any small adjustment may cause the filtering algorithm to fail. We solve this by having some reasonable starting values, and run the filtering algorithm repeatedly where the parameters is adjusted one by one, to lower the average log-likelihood (see line 41 in Algorithm 1 for the computation). This is a rather time consuming process, where an automatic Expectation Maximization (EM) approach would have been preferable.

Finally, the observation model is given as

$$\begin{aligned} \mathbf{O}^{(t)} \mid \mathbf{Y}^{(t)} &\sim \mathcal{N}(\mathbf{C}\mathbf{Y}^{(t)}, \mathbf{R}) \\ \mathbf{C} &= \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}, \quad \mathbf{R} = \operatorname{diag}(1.2256, 1.2256)^2, \end{aligned} \tag{4.42}$$

where Garmin (2022c) states that GPS location accuracy is around 3 meters 95% of the time. By solving

$$\iint_{\sqrt{x^2+y^2}<r} f(x; 0, R) f(y; 0, R) \, \mathrm{d}x \, \mathrm{d}y = 1 - \alpha,$$

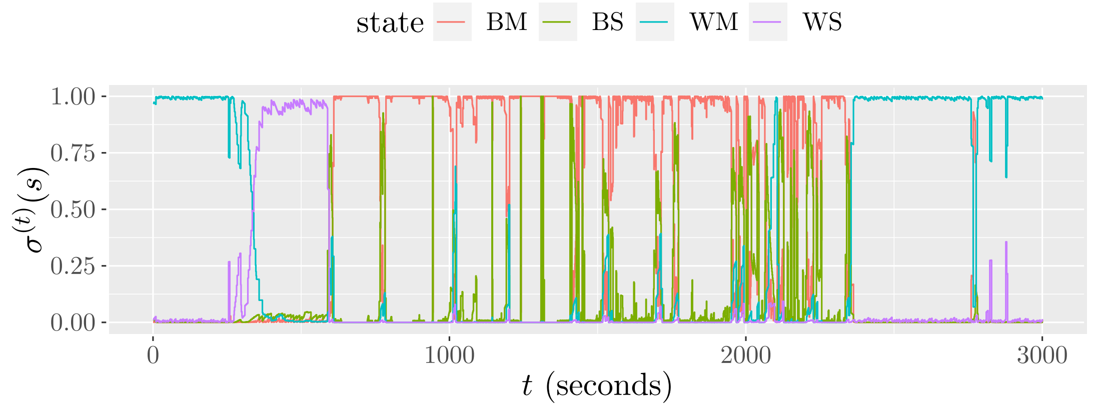
for R, we find that the variance must be

$$R = \operatorname{Var}[O_x^{(t)} \mid \mathbf{Y}^{(t)}] = \operatorname{Var}[O_y^{(t)} \mid \mathbf{Y}^{(t)}] = -\frac{r^2}{2 \log \alpha} = 1.2256^2, \tag{4.43}$$

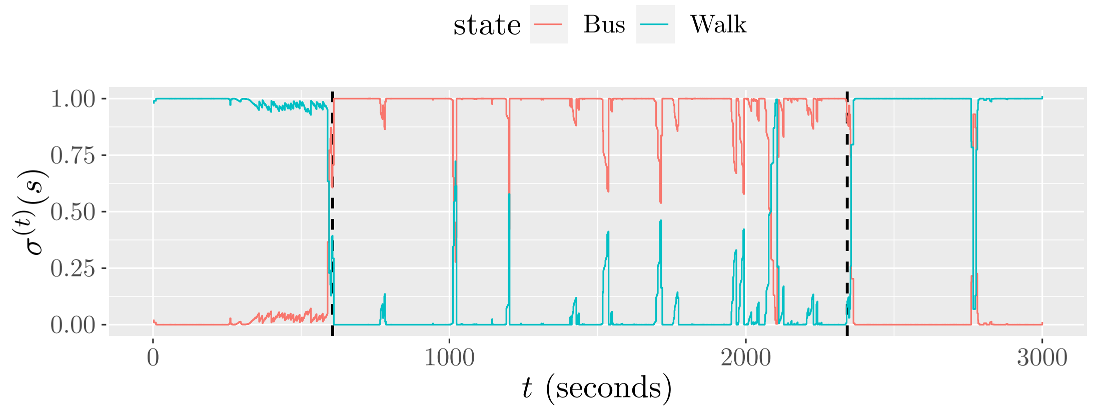
with significance level $\alpha = 0.05$ and radius $r = 3$ meters.

For the filtering, we choose to go with the IMM approximation Section 3.4.5 and 4.1 showed that it works just as well as GPB2, and is more efficient. The filtered estimate of the switching state is shown in Figure 4.8, where we see the marginal probabilities of all four states in Figure 4.8a and the joint probability of the "transport type" in Figure 4.8b. The filtered estimate correctly predicts whether or not we are on the bus, as the ground truth is already known and shown by the vertical dashed lines. We also see that the estimate is much less turbulent when in the walking states, compared to when on the bus, which is not surprising as the motion of the bus is much more complex. Figure 4.9a shows the filtered estimates of WS and WM the first ten minutes of the trip. When stopping, it takes about one and a half minute before the probability of being in state WS surpasses the probability of being in state WM. When entering the bus after about 10 minutes into the trip, both probabilities drops to zero. Figure 4.9b shows the filtered estimates of BS and BM at the beginning of the trip when entering the bus. We see that the estimate performs well and correctly classifies when the bus is stopping and when it is moving, with some small disturbances. Taking a look again at Figure 4.8a, at around 2000 seconds into the trip, the filtered estimate fluctuates much more frequently, and at around 2100 wrongfully predicts the state WM for a few seconds. This is when the bus is in Trondheim city center, which contains a lot of traffic lights and where the traffic moves at a crawl.

The filtered estimate of the dynamic state is shown in Figure 4.10. Some interesting observations here are the yaw angle, velocity and acceleration. The yaw angle ranges from -10 to 20 radians, which indicates a range of more than four revolutions in the trajectory has taken place, which is incorrect. Also, we show the absolute value of the velocity although it was negative a large part of the time. The data retrieved from Garmin also contains an estimate of the velocity which is computed from sensors of the smartwatch. Wee see that our estimate of the velocity matches Garmin's estimate quite well, especially in the beginning. From around at $t = 1000$ seconds our estimate is quite

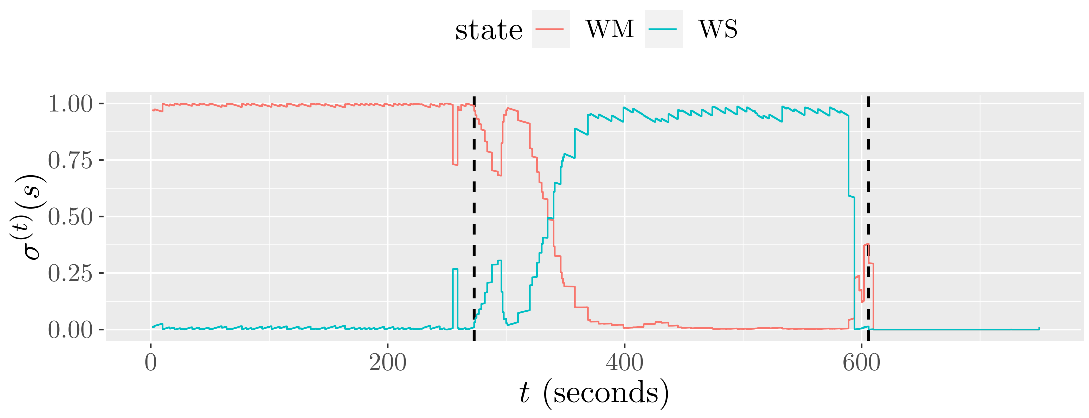


(a) The filtered estimate of the switching state.

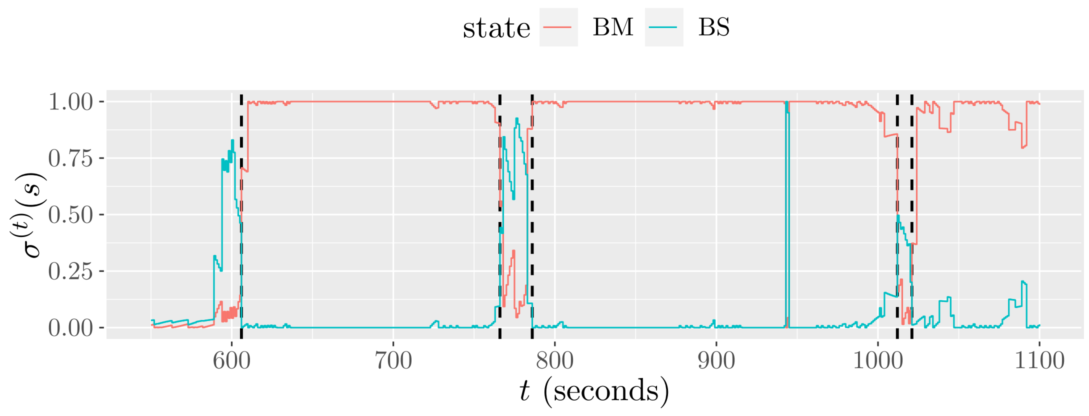


(b) Filtered estimate we combined WS with WM and BS with BM. The vertical dashed lines indicate when the bus is actually entered and exited, respectively.

Figure 4.8: Filtered estimate of the switching state, where (a) shows the marginal probabilities and (b) shows the joint probabilities of the walk states, and bus states.



(a) Filtered estimate of the states when walking (WS and WM).



(b) Filtered estimate of the states when on the bus (BS and BM).

Figure 4.9: The filtered estimate of the switching state showing the probabilities of the walking states in (a) and the bus states in (b). The vertical dashed lines indicate when the actual state switches occur, i.e. when we are entering and leaving a bus stop.

noisy (peaking at 60 m/s or about 215 km/h), and the same for acceleration (peaking at 30 m/s^2). These outliers may be due to weak GPS signal when being inside the bus. As we mentioned in Section 4.1, negative or positive velocity in the model does not really matter, but the sudden big jump in the yaw angle is troubling.

Although we managed to correctly classify when being on the bus and not, there is certainly room for improvement to make the result of the dynamic state more realistic. We only assumed one moving state for the bus, but the motion of the bus is complex, and could possibly be broken down into more states, such as turning, accelerating and constant velocity, and combinations of these.

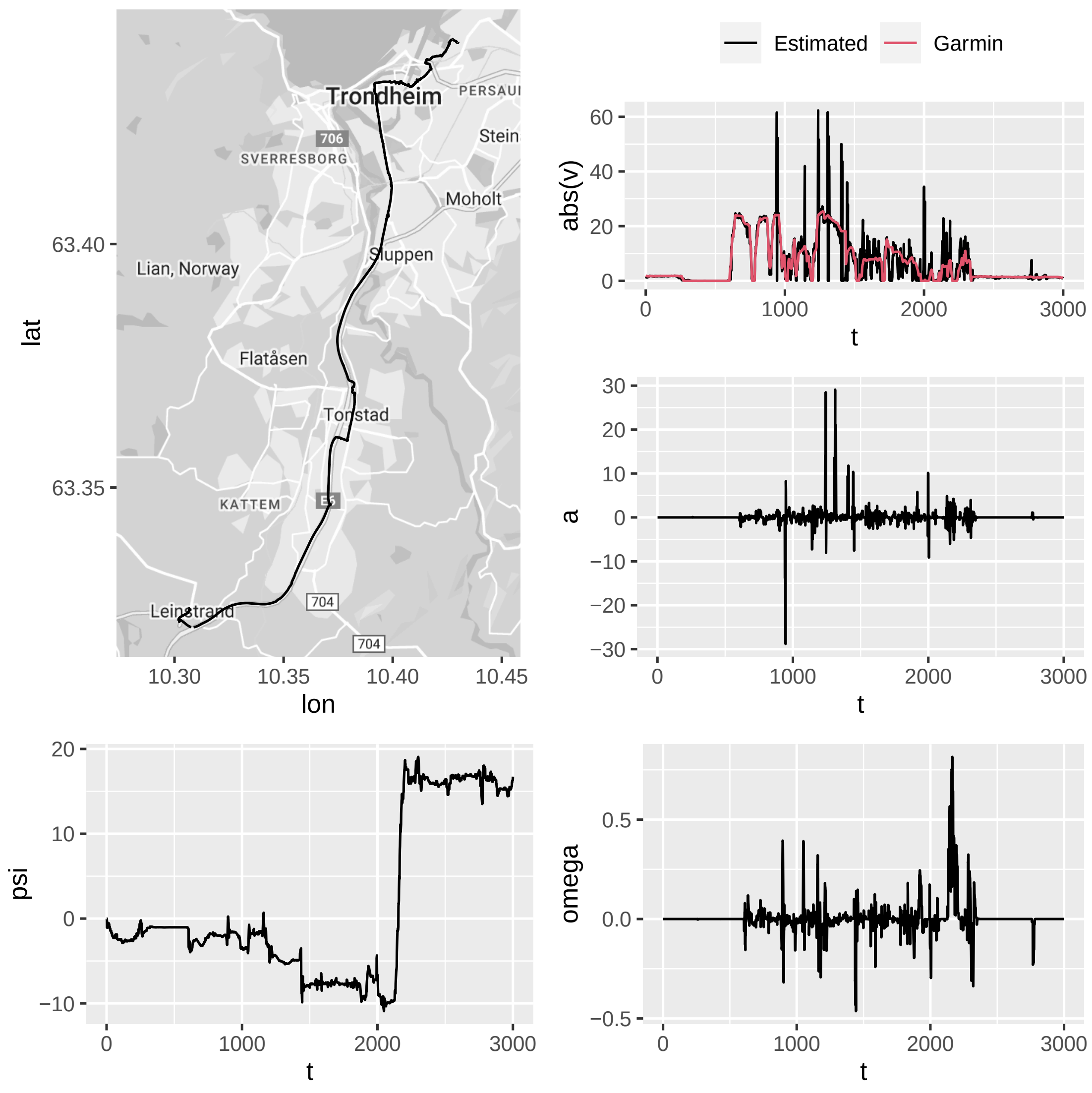


Figure 4.10: Filtered estimate of the dynamic state. The absolute value of the velocity estimate is shown instead of the original velocity estimate and is compared with Garmin's estimated velocity (red curve).

Chapter 5

Closing remarks

We have explored how we can use the language of graphs to model physical processes, where we use our intuition about causality to form dependence between variables. Then, we can define the CPDs where we condition all variables on their parent nodes. When performing inference on BNs, we saw how we can use the notion of d-separation to say that two variables are conditionally independent given another variable, to greatly reduce the complexity of the computations.

We have seen how we can introduce time to form DBNs where we used the state-observation model representation such as HMMs, LDS and SLDSs. For purely discrete cases such as the HMM, or pure continuous like the LDS, optimal inference is easy and efficient where we compute the filtered estimate recursively and exploit two important conditional independence properties, one in the prediction step, and the other in the condition step. However with SLDSs, we have seen that although we can use the same procedure and the same conditional independence properties, we get a problem with an exponential increase of mixture distributions, where each component in the mixture corresponds to a unique combination of the history of the switching states.

The main idea behind the GPB1, GPB2 and IMM approximations, is to collapse this increasing mixture after each time step in the recursive inference algorithm. GPB1 is the most gross approximation where we end up with a single Gaussian distribution after each time step, whereas for GPB2 and IMM we have a Gaussian mixture with the same size as the number of possible switching states. Although GPB2 is the most accurate approximation of the three, IMM appears to perform equally well and has a significantly lower computational cost, which makes IMM a favorable choice. The particle filter approach is very different and is based on weighted bootstrap sampling. It has the advantage that it is easy to implement but at the cost of a longer running time where it increases linearly as we increase the number of particles. We have seen that the particle filter scores the worst on accuracy of the switching state, both for 50 and 200 particles, even worse than the GPB1. There exists more specific particle filtering approaches that are likely to perform much better, but it was interesting to explore this simple general bootstrap approach.

We have seen how we can incorporate the EKF for nonlinear dynamical systems in

SLDSs, and we have used this where we tried to classify when we are on the bus, and when we are walking given GPS data. We have showed that for only four states the IMM successfully classifies when we are walking and when we are taking the bus, although we saw some strange results from the filtered estimate of the dynamic state, such as a sudden four-revolutions turn of the bus and a few outliers with a velocity of more than 200 km/h.

Some possible improvements that could be done in the model for the bus and walk example is to include an edge from the dynamic state to the switching state in the next time step. This way we can increase the probability of switching from an accelerating state to a constant velocity state as the velocity increases, and similarly increase the probability of switching from an accelerating state to a being stationary when the velocity is close to zero. Other improvements include increasing the number of states when the bus moves as the motion dynamics are quite complex. We used EKF to deal with the nonlinearity, but it would be interesting to see how the unscented Kalman filter performed for this problem. Although the parameters can often be set using expert knowledge, automatic approaches for learning the parameters is an important aspect of statistical modeling. The most tedious task for the bus and walk example was to estimate the the variances in the transition model as it was done manually in an expectation-maximization manner. Finally, we have seen that most of the covariance matrices are very sparse. We did not take advantage of that in the algorithm, so this could be a possible improvement to speed up the filtering running time, especially as length of the dynamic state vector increases.

We have only scratched the tip of the iceberg of SLDSs with the bus and walk example, but many real-world examples have similar set-ups such as object tracking for stop-and-go situations which are used for adaptive cruise control and collision avoidance. Also, expanding to allow for even more types of transports as a switching state, such as bicycle and personal car could potentially make navigation software more adaptive where it automatically “notices” when the type of transport switches, such that we can receive useful feedback on estimated time of arrival or whether to expect slow traffic.

Bibliography

Bar-Shalom, Y., Li, X. R., & Kirubarajan, T. (2004). *Estimation with applications to tracking and navigation: Theory algorithms and software*. John Wiley & Sons.

Blom, H. A., & Bar-Shalom, Y. (1988). The interacting multiple model algorithm for systems with Markovian switching coefficients. *IEEE transactions on Automatic Control*, *33*(8), 780–783.

Durlauf, S., & Blume, L. (2016). *Macroeconometrics and time series analysis*. Springer.

Garmin. (2022a, May). *About us*. https://www.garmin.com/en-US/company/about-garmin/

Garmin. (2022b, April). *The flexible and interoperable data transfer (FIT) for storing and sharing data*. https://developer.garmin.com/fit/

Garmin. (2022c, April). *What can cause GPS accuracy issues on my fitness device?* https://support.garmin.com/en-US/?faq=z0n0KE1XVF0Pe4Su8QiZgA

Geonorge. (2022, April). *Brukerveiledning*. https://www.geonorge.no/aktuelt/om-geonorge/brukerveiledning/#Formater

Gu, Y., & Raphael, C. (2012). Modeling piano interpretation using switching Kalman filter. *ISMIR*, 145–150.

Julier, S. J., & Uhlmann, J. K. (1997). New extension of the Kalman filter to nonlinear systems. *Signal processing, sensor fusion, and target recognition VI*, *3068*, 182–193.

Kaempchen, N., Weiss, K., Schaefer, M., & Dietmayer, K. (2004). IMM object tracking for high dynamic driving maneuvers. *2004 IEEE Intelligent Vehicles Symposium*, 825–830.

Kahle, D., & Wickham, H. (2013). Ggmap: Spatial visualization with ggplot2. *The R Journal*, *5*(1), 144–161. https://journal.r-project.org/archive/2013-1/kahle-wickham.pdf

Kalman, R. E. (1960). A new approach to linear filtering and prediction problems.

Koller, D., & Friedman, N. (2009). *Probabilistic graphical models: Principles and techniques*. The MIT Press.

Lerner, U. N. (2003). *Hybrid Bayesian networks for reasoning about complex systems*. Stanford University Press.

Li, N., & Stephens, M. (2003). Modeling linkage disequilibrium and identifying recombination hotspots using single-nucleotide polymorphism data. *Genetics*, *165*(4), 2213–2233.

Meuter, M., Muller-Schneiders, S., Mika, A., Hold, S., Nunn, C., & Kummert, A. (2009). A novel approach to lane detection and tracking. *2009 12th International IEEE Conference on Intelligent Transportation Systems*, 1–6. https://doi.org/10.1109/ITSC.2009.5309855

Microsoft Docs. (2022, April). *WGS84 class.* https://docs.microsoft.com/en-us/previous-versions/windows/embedded/cc510650(v=msdn.10)?redirectedfrom=MSDN#remarks

Murphy, K. P. (1998). *Switching Kalman filters* (tech. rep.). DEC/Compaq Cambridge Research Labs.

Murphy, K. P. (2002). *Dynamic Bayesian networks: Representation, inference and learning.* University of California, Berkeley.

Pearl, J. (1988). *Probabilistic reasoning in intelligent systems: Networks of plausible inference* (1st ed.). Morgan Kaufmann.

Pebesma, E. (2018). Simple Features for R: Standardized support for spatial vector data. *The R Journal*, *10*(1), 439–446. https://doi.org/10.32614/RJ-2018-009

Rabiner, L. R. (1989). A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, *77*(2), 257–286.

Ravindranath, V. K. (2019, February). *Finding Dory, hidden Markov models and simplifying life.* https://towardsdatascience.com/finding-dory-hidden-markov-models-and-simplifying-life-3a69f01b4d50

Rudenko, A., Palmieri, L., Herman, M., Kitani, K. M., Gavrila, D. M., & Arras, K. O. (2020). Human motion trajectory prediction: A survey. *The International Journal of Robotics Research*, *39*(8), 895–935.

Russell, S., & Norvig, P. (2021). *Artificial intelligence: A modern approach, global edition* (4th ed.). Pearson Education.

Särkkä, S. (2013). *Bayesian filtering and smoothing.* Cambridge University Press.

Scott, S. L. (2002). Bayesian methods for hidden Markov models: Recursive computing in the 21st century. *Journal of the American statistical Association*, *97*(457), 337–351.

Starner, T., & Pentland, A. (1997). Real-time american sign language recognition from video using hidden Markov models. *Motion-based recognition* (pp. 227–243). Springer.

Vahidi, A., & Eskandarian, A. (2003). Research advances in intelligent collision avoidance and adaptive cruise control. *IEEE transactions on intelligent transportation systems*, *4*(3), 143–153.

West, M., & Harrison, J. (2006). *Bayesian forecasting and dynamic models.* Springer Science & Business Media.

Zucchini, W., & MacDonald, I. L. (2009). *Hidden Markov models for time series: An introduction using r.* Chapman; Hall/CRC.