Machine learning for modelling urban dynamics

Kira Kempińska

A dissertation submitted in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

of

University College London.

Department of Security and Crime Science

University College London

April 12, 2019

I, Kira Kempińska, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the work.

Abstract

We live in the age of cities. More than half of the world's population live in cities and this urbanisation trend is only forecasted to continue. To understand cities now and in the foreseeable future, we need to take seriously the idea that it is not enough to study cities as sets of locations as we have done in the past. Instead, we need to switch our traditional focus from locations to interactions and in doing so, invoke novel approaches to modelling cities.

Cities are becoming "smart" recording their daily interactions via various sensors and yielding up their secrets in large databases. We are faced with an unprecedented opportunity to reason about them directly from such secondary data. In this thesis, we propose model-based machine learning as a flexible framework for reasoning about cities at micro and macro scales. We use model-based machine learning to encode our knowledge about cities and then to automatically learn about them from urban tracking data. Driven by questions about urban dynamics, we develop novel Bayesian inference algorithms that improve our ability to learn from highly complex, temporal data feeds, such as tracks of vehicles in cities. Overall, the thesis proposes a novel machine learning toolkit, which, when applied to urban data, can challenge how we can think about cities now and about how to make them "smarter".

Impact Statement

The research contributions presented in the thesis could be put to a beneficial use both inside and outside academia.

Inside academia, the research could benefit the domains of urban modelling and probabilistic machine learning. Firstly, urban modellers could use our adaptations of machine learning methods to spatio-temporal data. In fact, code packages developed and open-sourced during the thesis have already been used by a number of MSc and PhD students in spatial data science at UCL and beyond for data preprocessing and modelling. Secondly, the thesis develops novel machine learning algorithms that have already received interest from researchers in the machine learning domain of Bayesian deep learning (Spotify award winner at NIPS 2017).

Outside academia, the thesis could benefit both public and commercial entities. The thesis has been developed in close collaboration with the London Metropolitan Police, who have provided data for this research and have integrated some of our methods of map-matching into their internal patrol monitoring software. The thesis also has a clear commercial value that could be realised inside an existing business or as a UCL spin-out. In particular, the technology of map-matching could benefit a range of location-based services, such as taxis or deliveries, by improving the accuracy of location data in cities. As an example, Uber has developed a very similar technology for daily optimisation of their food delivery services. Our research presents ready-to-use methods for improved localisation outdoors. It could be further extended to localisation indoors, which could benefit a range of commercial and public bodies, from technologies for the blind to navigation apps inside large shopping malls.

Acknowledgements

First, I would like to thank my supervisors John Shawe-Taylor and Paul Longley. My first supervisor John Shawe-Taylor for his continuous support, incredible knowledge, openness to new ideas, but also for setting high expectations. His regular feedback and encouragement have shaped this thesis. My second supervisor Paul Longley for his continuous help, academic guidance and domain expertise.

I would also like to thank my parents and my husband Michał for their encouragement and patience as I have been immersed in my research. This thesis would not have been possible without their support. I am also grateful to my daughters Tosia and Iga for offering much needed distractions and enriching life with more than just research.

The research presented in this thesis is part of the project - Crime, Policing and Citizenship (CPC): Space-Time Interactions of Dynamic Networks (www.ucl.ac.uk/cpc), supported by the UK Engineering and Physical Sciences Research Council (EP/J004197/1). The data provided by Metropolitan Police Service (London) are greatly appreciated.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

We live in an inherently urban world. A world where more people live in cities than in the countryside and around seventy per cent of global gross domestic product is generated in cities. If we want to understand where we are and foresee our future, we must turn our attention to understanding cities.

Cities are not merely spaces and places, but rather systems of networks and flows. As Batty claims in his recent book ”The New Science of Cities” [1], in order to understand cities now and foresee their evolution into the future, we must view cities as ”places where people come together to ’interact’ with one another”. These interactions define cities now and give an indication of where they are heading. As new technologies influence the way we interact, they also create a detailed record of our interactions. Cities are becoming ”smart”, recording their daily pulses via various sensors and yielding up their secrets in the form of very large databases. We are presented with an unprecedented opportunity to build a detailed understanding of cities directly from data.

The purpose of this thesis is to develop machine learning algorithms that enable us to reason about cities directly from urban sensor data. Machine learning is the science of learning from data. It gives us a toolkit to automatically detect patterns in data, and then use the discovered patterns to reason about cities, both in terms of what they are now and in terms of how to make them ”smarter”. A key concept in machine learning is that of uncertainty. It can arise through noise in measurements, limited size of data sets, or uncertainty about the best model to explain data. In

this thesis, we adopt the view that the best way to solve such problems is to use the tools of probability theory. Probability theory provides a consistent framework for the quantification and manipulation of uncertainty and forms the foundation for probabilistic machine learning. It allows us to make optimal predictions and decisions even if the data we hold on cities may be incomplete or noisy.

Cities are composed of multiple types of interactions. In the thesis, we limit our focus to ”physical” interactions as captured in large volumes of tracking data. The dynamics take place in space and time and are constrained to the structure of the urban road network. At the core of the thesis, we propose novel machine learning methods that enable researchers to derive micro and macro-level insights into city dynamics directly from the spatio-temporal tracking data.

1.1 Key contributions

The thesis proposes model-based machine learning, fulfilled as Bayesian inference in probabilistic graphical models, as a flexible and principled framework for reasoning from urban data. Driven by questions on urban dynamics, we advance the current state-of-the-art in Bayesian inference in probabilistic graphical models. In particular, our research is motivated by the following questions:

1. **Where you are**: can we infer true locations from noisy tracking data?

2. **How you navigate**: can we extract urban regions, as perceived by drivers, from their tracking data?

1.1.1 Where you are

To address the question of *where you are*, we propose three model-based algorithms for map-matching, i.e. inferring locations on the road network given noisy tracking data:

- **probabilistic ST-Matching**: a probabilistic adaptation of ST-Matching algorithm [2]; preserving high computational efficiency of ST-Matching, while equipping it with the ability to express map-matching confidence;

- **particle filter**: a probabilistic approach to map-matching; offering higher map-matching accuracy at the expense of higher computational cost;

- **look-ahead particle filter**: an adaption of particle filter that leads to order of magnitude improvements in accuracy and computational efficiency when tracking data are sparse.

1.1.2 How you navigate

To answer the question of *how you navigate*, we propose adaptations of a probabilistic *topic* model, called latent dirichlet allocation, to extracting urban regions as *topics* from vehicle tracking data. The obtained regions capture spatial routing preferences, hence they enable accurate route prediction.

1.1.3 Methodological extensions

Finally, we propose novel algorithms for Bayesian inference in probabilistic graphical models which extend beyond the motivating questions on urban dynamics.

- **adversarial sequential Monte Carlo**: an importance sampling algorithm for approximate inference which borrows ideas from generative adversarial networks (GANs) to train highly flexible proposal models;

- **moment matching variational Bayes**: an improved variational autoencoder that learns a highly expressive inference model using an idea from statistical hypothesis testing known as moment matching.

1.2 Thesis outline

The thesis begins with an introduction to urban dynamics and to the motivating case study of police urban dynamics in Chapter 2.

Chapter 3 introduces model-based machine learning as the proposed toolkit for modelling urban dynamics. It describes the basic building blocks of model-based machine learning: graphical models and approximate inference. Finally, it brings to focus approaches to modelling sequential data as particularly relevant to modelling

urban dynamics. The following two chapters develop model-based approaches to modelling urban dynamics using tracking data.

Chapter 4 proposes model-based approaches to improved localisation in cities. It introduces three novel methods of map-matching, that is, improved localisation by aligning tracking data with underlying road networks. This research is particularly useful in cities, where high rise buildings cause large errors in satellite-based localisation.

Chapter 5 suggests data-driven tools for urban regionalisation. The approaches extract regions directly from urban tracking data. The regions capture routing preferences of drivers in cities and, as such, can be used for improved route prediction.

Chapter 6 presents methodological innovations that extend beyond the initial application of urban dynamics. In particular, it demonstrates novel approaches to approximate Bayesian inference that use deep generative models to train highly flexible inference models. This research contributes new ideas in Bayesian deep learning that are applicable to a range of problems, with urban modelling being one example.

The thesis concludes by reviewing the key contributions and directions for future work. It critically evaluates the presented research and discusses future work that could address its shortcomings or further extend or validate its contributions.

1.3 Publications

Parts of the thesis have been published in a number of journal and conference proceeding papers. Below is a list of the publications grouped by the chapter that they correspond to.

Chapter 4: Where you are

1. Kira Kempinska, Toby Davies, John Shawe-Taylor, and Paul Longley. Probabilistic map-matching for low-frequency gps trajectories. In *Proceedings of GIS Ostrava conference*, pages 209–221. Springer, 2017. This paper features the probabilistic ST-Matching algorithm.

2. Kira Kempinska, Toby O. Davies, and John Shawe-Taylor. Probabilistic map-matching using particle filters. In *Proceedings of the 24th GIS Research UK conference*, 2016. This paper described the basic implementation of particle filters for probabilistic map-matching.

3. Kira Kempinska and John Shawe-Taylor. Improved particle filters for vehicle localisation. In *NIPS 2016 Advances in Approximate Inference workshop*, 2016. This paper introduced the look-ahead particle filters algorithm.

Chapter 5: How you navigate

1. Kira Kempinska, Paul Longley, and John Shawe-Taylor. Interactional regions in cities: making sense of flows across networked systems. *International Journal of Geographical Information Science*, 32(7):1348–1367, 2018. This journal paper describes our methodology for extracting interactions regions from vehicle tracking data.

2. Kira Kowalska, John Shawe-Taylor, and Paul Longley. Data-driven modelling of police route choice. In *Proceedings of the 23rd GIS Research UK conference*, 2015. The early paper explores the use of interactional regions for route prediction.

3. T Cheng, Kate Bowers, Paul Longley, John Shawe-Taylor, Trevor Adams, Toby Davies, Gabriel Rosser, Sarah Wise, Chris Gale, Monsuru Adepeju, Jianan Shen, Huanfa Chen, Dawn Williams, Kira Kempinska, and Artemis Skarlatidou. *CPC: Crime, Policing and Citizenship - Intelligent policing and big data.* 05 2016. This book includes our work on police route choice modelling.

Chapter 6: Methodological extensions

1. Kira Kempinska and John Shawe-Taylor. Adversarial sequential monte carlo. In *NIPS 2017 Advances in Approximate Inference workshop*, 2017. Our most recent work that won the contributed talk and the Spotify award.

Chapter 2

Urban dynamics

> Moving elements in a city, and in particular the people and their activities, are as important as the stationary physical parts.
>
> Kevin Lynch

2.1 Urban dynamics as networks and flows

Jane Jacobs once said, ”Cities have the capability of providing something for everybody, only because, and only when, they are created by everybody” [3]. In this statement, Jacobs makes a clear point that cities are composed of people and their relations, communications, interactions. If we want to understand cities and foresee their evolution in time, we should view them as complex systems of interacting individuals rather than simply locations. This idea lays a foundation to the new science of cities recently proposed by Batty [1], in which he claims that cities are constellations of interactions and that locations are important, but only as places which anchor the interactions. Locations are a spatial manifestation of various processes and interactions, which bring people together to produce and exchange goods and ideas. Instead of thinking of cities as sets of spaces or locations, we need to think of them as sets of actions, interactions, and transactions that are rooted in space and time.

The key to understanding cities, then, is the way we unpick their physical form

to reveal the interactions that define them. At the core of the new science of cities, we abstract the interactions that take place in cities as systems of flows and networks. We use flows to denote interactions and networks to represent structures on which flows take place.

Flows and networks are essentially two sides of the same coin. Flows exist between any two locations and are not necessarily embedded in the physical structure of the urban space, but can float somehow freely across space. They are an abstraction of processes that operate between locations. For example, they can represent journey to work patterns between two locations as a function of their accessibilities and potential. Networks are more closely embedded in the physical urban structure. They are the ”physical containers” whose capacity constrains flows of energy and information, people or goods between locations. Their nodes represent locations, but they are not necessarily fixed. For example, airline networks might have nodes fixed at specific locations, whereas social networks might have nodes and links that are constantly moving in space. All in all, all networks considered in urban science have some spatial association in that they always pertain to cities and their urban form.

Cities comprise of a multitude of networks and flows with varying level of embeddedness in space. This thesis is limited to the strictest example of networks, whose nodes and links are fully embedded in space, that is, street networks. Invariably, street networks carry flows that represent the physical facet of interactions in cities. The goal of this thesis is to propose tools which build our understanding of cities from the *physical* flows on street networks, as captured in large volumes of vehicle tracking data.

2.2 Challenges in modelling urban flows

Urban flows of vehicles generate an abundance of tracking data. If properly analysed, *machine learning*, the science of learning from data, can turn the data into a detailed understanding of urban dynamics.

Machine learning has benefited from several decades of research during which

a multitude of successful algorithms have been developed, each designed to model a particular problem. When a researcher attempts to apply machine learning to tackle their problem, they typically try to map it to one of the 'off-the-shelf' algorithms. The success of such an approach is largely dependent on the availability of suitable algorithms.

The suitability of a machine learning algorithm for modelling urban dynamics depends on assumptions that the algorithm makes about data. Most machine learning algorithms have been designed under the i.i.d., independent and identically distributed, assumption that your data points are independently collected and represent the same process being modelled. Although the assumption is a useful simplification in algorithmic design, it is not applicable to vehicle dynamics. Vehicle tracking data is inherently spatiotemporal in the sense that data points collected in a close temporal and spatial proximity are interdependent. If we want to gain correct insights into urban dynamics, we should account for the data interdependence in the modelling process. In the thesis, some of the key methodological contributions are adaptations of existing machine learning algorithms to derive insights from *spatiotemporal* data.

Urban flows of vehicles happen on a very small subset of space, namely, the road network (see Figure 2.1 for an example). Another key methodological contribution in the thesis is the development of machine learning techniques which can use the prior knowledge of the structure of the road network to refine the data-driven insights into city dynamics.

2.3 Our case study

The proposed methodology is validated through the use of tracking data from police patrol vehicles. Unlike other urban activities, such as mail delivery or shopping, police patrolling is expected to take place in every part of the city. The granularity of spatial coverage makes this data particularly suitable for deriving both micro- and macro-level insights into urban dynamics. Although the derived insights might at times be specific to the way police navigate around cities, the proposed method-

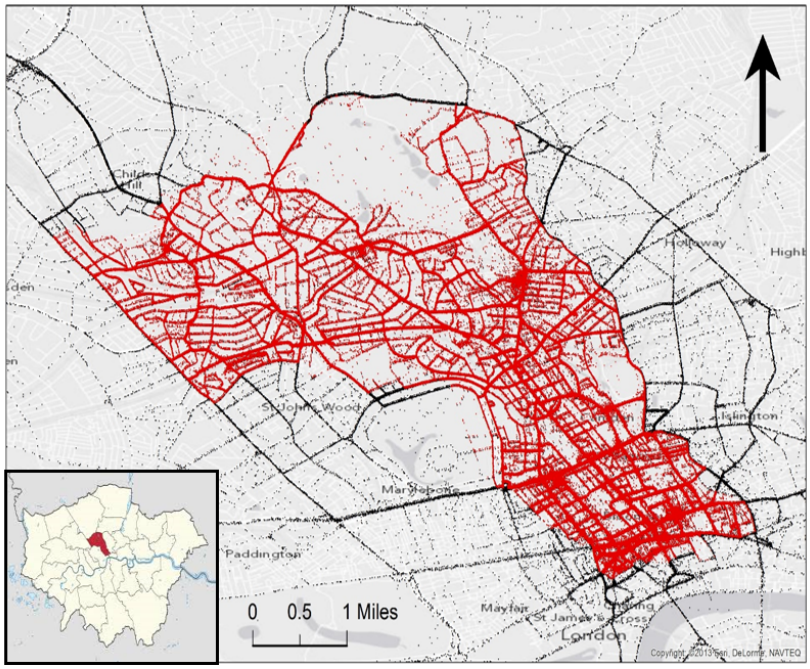


Figure 2.1: GPS signals transmitted by police vehicles inside (red) and outside (black) the London Borough of Camden in March 2011.

ology is agnostic to activity type and can be equally applied to any other vehicle tracking data.

The police patrol data are a complete set of GPS signals transmitted by police patrol vehicles during March 2011 in the London Borough of Camden, a borough in Central London with the total area of 21.8 km^2. The dataset comprises a total of 1,188,953 GPS signals from 5,513 journeys (see Figure 2.1). It was acquired for research purposes as part of the "Crime, Policing and Citizenship" project in collaboration with the Camden Metropolitan Police[1].

[1]UCL Crime Policing and Citizenship: http://www.ucl.ac.uk/cpc/.

Chapter 3

Model-based machine learning

> As far as the laws of mathematics refer to reality, they are not certain, as far as they are certain, they do not refer to reality.
>
> Albert Einstein

Most tasks require a person or an automated system to *reason*: to take the available information and reach conclusions, both about what might be true in the world and about how to act. In the context of smart cities, for example, a computer program needs to take video footage and recognise free parking spots to direct cars entering the car park. An autonomous vehicle needs to synthesise data from its positioning sensors, cameras and other sensors to conclude where it is and how to move to reach its goal without hitting anything.

In this thesis, we use a model-based framework to answer questions of this type. In principle, one could write a special-purpose computer program for every domain and question that one may wish to answer. The resulting program, although possibly quite successful at its particular task, requires effort and is often very brittle: if the application changes, significant changes to the programme may also be required. Alternatively, one could use one of the 'off-the-shelf' programs to answer their question, their success would be largely dependent on the availability of suitable models. This approach is quite limiting: it is hard to extract lessons from one successful solution and then apply it to another quite different problem.

We focus on a different approach, based on the concept of a *declarative representation*. In this approach, we construct a *model* of the system that we want to reason about. The model encodes our knowledge of how the system works. We then apply various algorithms to the model to answer questions about our system given observed data. For example, a model might represent our knowledge about unexpected traffic conditions, such as road closures or car accidents, and how they impact on journey times. A reasoning algorithm can take this model, along with real-time traffic data, to reason if any unexpected traffic conditions have occurred and update estimated travel times accordingly. The key concept in model-based learning is the separation of knowledge and reasoning. The model representing knowledge is separate from the algorithms that one can apply to it. Thus, we can develop general purpose reasoning algorithms that apply to any model, whether in the domain of traffic modelling or computer vision. Conversely, we can create highly tailored models for specific scenarios, as well as rapidly prototype and compare a range of alternative models without the need to modify our reasoning algorithms constantly. The combination of the model and the inference procedure together define a machine learning algorithm, as illustrated in Figure 3.1.

There could be many ways of fulfilling the vision of model-based learning. In this thesis, the focus is on a powerful framework based on Bayesian inference and probabilistic graphical models. The framework is grounded in probability theory, which enables us to reason about cities under significant amounts of uncertainty. Uncertainty is an inescapable aspect of most real-world applications, including city dynamics, and it stems from several factors. We are uncertain about the true state of the system because are observations are partial and noisy. For example, traffic conditions are never directly observed and can only be inferred from noisy GPS data. Our observations can be consistent with many models, hence we might also be uncertain about the most appropriate model for our system. Finally, the system might be nondeterministic by nature, hence we can never be absolutely certain about its state. All in all, uncertainty plays a fundamental role in our ability to reason about systems, such as cities, and calls for a systematic treatment in a modelling

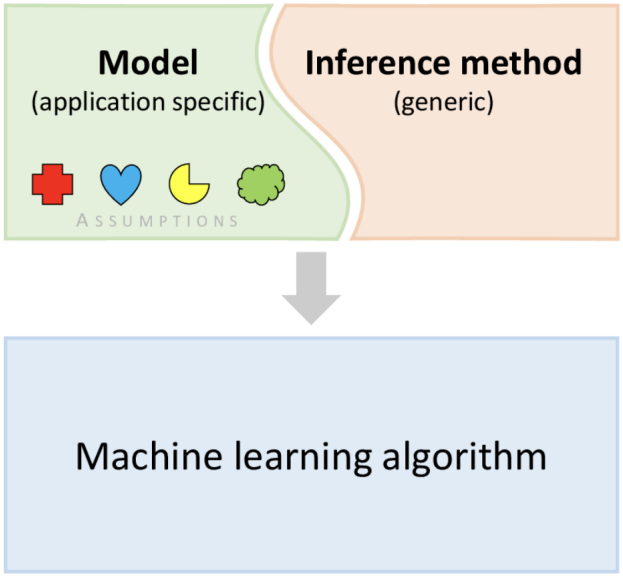


Figure 3.1: In the model-based view of machine learning, a machine learning algorithm arises from a particular combination of a model and an inference method. Here the coloured shapes within the model represent the assumptions comprising that specific model. Changes to the assumptions give rise to different machine learning algorithms, even when the inference method is kept fixed. Source: [4].

process. Bayesian inference offers a framework for reasoning about real-world systems under uncertainty and has been successfully used in a range of applications from predicting journey times [5] to discovering new planets [6].

3.1 Representing uncertainty

The key idea behind model-based machine learning is that learning can be thought of as ”inferring plausible models to explain observed data” [7]. Observed data can be consistent with many models, and therefore most appropriate model for the data might be uncertain. Observations might be noisy in which case uncertainty might also be inherent to the data themselves. All in all, uncertainty plays a fundamental role in learning from data and thus calls for a systematic treatment in a modelling process.

Probability theory lays the foundation for representing uncertainty in modelling. It is conceptually simple, yet capable of expressing all forms of uncertainty: uncertainty about the data (due to noise etc.), uncertainty about the structure of an appropriate model and the values of its parameters. The probabilistic approach

underlie two simple rules:

- Sum rule: $p(x) = \sum_{y \in Y} p(x,y)$
- Product rule: $p(x,y) = p(x)p(y|x)$

where $y \in Y$ are non-overlapping events. At a conceptual level, it is easy to argue why the probabilistic approach is *the* way to represent uncertainty in models. The Cox axioms provide a list of conditions for representing beliefs; a consequence of these axioms is that 'degrees of belief' must follow all the rules of probability theory [8, 9, 10]. This justifies using Bayesian probabilities when modelling with uncertainty. An additional argument comes from decision theory. The Dutch book theorem states that unless an artificial intelligence system's (or human's, for that matter) degrees of belief are consistent with the rules of probability, it will be willing to accept bets that are guaranteed to lose money [11]. Therefore, if we want to reason based on uncertain outputs of our models, Bayesian probability ensures that we do it right.

Model-based machine learning uses probability distributions to represent uncertainty about unobserved quantities in a model (structural, parametric and noise related) and their relationship to the data. It then uses a corollary of the sum and product rules, Bayes Rule, to infer the unobserved quantities given the data. Bayes Rule performs the inference by transforming our prior beliefs (defined before observing the data) and the data into posterior distributions over the unobserved quantities. The basic form of Bayes Rule is:

$$p(x|y) = \frac{p(y|x)p(x)}{p(y)} = \frac{p(y|x)p(x)}{\sum_{x \in X} p(y,x)} \tag{3.1}$$

In the context of model-based machine learning, we replace y by D to denote the observed data, we replace x by θ to denote the unknown parameters or latent variables, and we condition all terms on m, the model that we are considering. The resulting Bayes Rule calculates $p(\theta|D,m)$, the posterior of the unobserved quantities in the model given data:

$$p(\theta|D,m) = \frac{p(D|\theta,m)p(\theta|m)}{p(D|m)} \tag{3.2}$$

When defined at the level of m, Bayes Rule can also be used to compare different models:

$$\begin{aligned} p(m|D) &= \frac{p(D|m)p(m)}{p(D)} \\ p(D|m) &= \int p(D|\theta,m)p(\theta|m)d\theta \end{aligned} \tag{3.3}$$

The fact that model-based machine learning is fully probabilistic brings a number of advantages. As previously mentioned, it allows us to learn under uncertainty. Its basic building blocks, probability distributions, can be combined to build more complex, yet easily understandable models (see next section for more details). Its probabilistic toolkit, Bayes Rule in particular, provides a systematic way of model comparison (see Equation 3.3) that, unlike optimisation-based techniques, is not prone to over-fitting (see Bayesian Ockham's razor [12, 13, 14]).

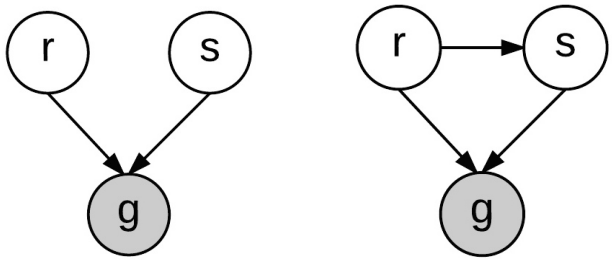
3.2 Building blocks

Model-based machine learning is a powerful framework that enables us to create custom reasoning engines for any domain. In this section, we introduce in depth its two main building blocks (see Figure 3.1): the *model* of the system that we want to reason about and the *inference* method that we can apply to the model to reason from data.

3.2.1 Graphical models

In a Bayesian setting, a 'model' is built from simple probability distributions over a single or a few variables that are composed to form more complex model structures. The dominant paradigm for representing such compositional probabilistic models is graphical models.

Graphical models give us a pictorial representation of the relationships among model variables, possibly embodying prior beliefs or knowledge about causal relationships. They express how the joint distribution over all the random variables in



(a) independent **(b)** dependent

Figure 3.2: Simple graphical models representing rain r and sprinkler s as independent or dependent causes of grass g being wet. According to common practice, observed variables are grey, unobserved variables are white.

the problem is factored into the product of distributions over smaller sets of variables. If we denote by x the unobserved (latent) variables and by y the observed data, the factorisation is such that:

$$p(x,y) \triangleq p(x)p(y|x) \tag{3.4}$$

The factorisation is what later dictates techniques for learning with such models.

The easiest way to interpret the graphical representation is through the so-called generative viewpoint; we can imagine that observed data were generated by sampling from the graph. In fact, it is common practice to generate synthetic data from the graph to ensure that the dependencies between variables are correct.

Let us illustrate graphical models with an example. Imagine that we want to build a model to represent possible causes of the grass (denoted by variable g) being wet or dry. The grass is wet if it rains (r) or if the sprinkler (s) is activated. In the notation proposed above, we have latent variables $x \equiv \{r,s\}$ and observed variable $y \equiv g$ .We can assume that the activation of the sprinkler is independent of rain (Figure 3.2a) or not (Figure 3.2b), leading to different factorisations of the joint distribution over all the variables $p(g,r,s)$:

- Independent causes: $p(g,r,s) = p(g|r,s)p(s)p(r)$
- Dependent causes: $p(g,r,s) = p(g|r,s)p(s|r)p(r)$

Two powerful aspects of graphical models have been illustrated by the example. Firstly, note that the graphical models have not required specifying whether the variables are discrete or continuous, nor the functional form of their relationships, such as Gaussian, Bernoulli or gamma distributions. The factorisations are therefore very general and apply to a whole family of models. Secondly, it has been very easy to incorporate an additional dependency between variables. This shows the flexibility of a graphical model to be tailored to a specific application, or modified if the requirements of the application change.

3.2.2 Approximate inference

A graphical model is just one side of the coin in model-based machine learning. What we are still missing is the ability to learn about unobserved quantities in the model given data. This task is called *inference*. It requires calculating the posterior distribution over unobserved quantities in the model given the data (Equation 3.2). In mathematical terms, inference is the evaluation of the posterior distribution $p(x|y)$ of the unobserved (latent) variables x given the observed data variables y. According to Bayes Rule (previously introduced in 3.1), the calculation is equal to

$$p(x|y) = p(y|x)p(x) \Big/ \int p(y,x)dx \tag{3.5}$$

Although conceptually simple, the calculation is infeasible for many real-life models because of high dimensional integration over the latent space or a highly complex form of the posterior distribution itself.

In such situations, one must resort to approximate inference schemes. These fall broadly into two classes, according to whether they rely on stochastic or deterministic approximations. Stochastic techniques, such as Markov chain Monte Carlo or sequential Monte Carlo [15], approximate the posterior with a finite number of samples. They can provide an arbitrarily accurate approximation depending on the number of samples they produce. In practice, sampling methods can be computationally demanding, thus limiting their applicability to small-scale problems. It can also be difficult to know whether a sampling scheme produces independent sam-

ples from the required distribution. On the other end of the spectrum, deterministic approximation schemes are based on analytical approximations to the posterior distribution, for example by assuming that it factorizes in a particular way or that it has a specific parametric form such as a Gaussian [16]. They often scale better to large application but fail to ever generate exact results. These properties are in some sense complimentary to those of sampling methods.

3.2.2.1 Variational inference

Variational inference is a popular class of deterministic approximation algorithms that treats inference as an optimisation problem. The basic idea is to pick an approximation $q(x)$ from some tractable family, and then try to make this approximation as close as possible to the true posterior, $p^*(x) \triangleq p(x|y)$, by minimising the Kullback-Leibler (KL) divergence from p^* to q. That is, we wish to find q^* from a family of distributions Q, such that

$$q^* = argmin_{q \in Q} KL(q||p^*) \tag{3.6}$$

where the KL divergence is defined as

$$KL(q||p^*) = \int q(x) \log \frac{q(x)}{p^*(x)} dx \tag{3.7}$$

Unfortunately, Equation 3.7 is not tractable as written, since it requires evaluating the intractable distribution $p^*(x) = p(x|y)$. However, we can rewrite the KL divergence in terms of the likelihood and the so-called "evidence lower bound" (ELBO),

$$\begin{aligned} KL(q||p^*) &= \int q(x) \log \frac{q(x)}{p^*(x)} dx \\ &= \int q(x) \log \frac{q(x)p(y)}{p(x,y)} dx \\ &= \log p(y) + \int q(x) \log \frac{q(x)}{p(x,y)} dx \end{aligned} \tag{3.8}$$

.

Since the likelihood term $\log p(y)$ is a constant, by minimising ELBO, we will force q to become close to p^*.

Naturally, the quality of the learning depends on the expressiveness of the approximate inference model $q(x)$. One of the most popular forms of variational inference is the mean-field variational approximation [17], where $q(x)$ comes from a family of fully factorized distributions

$$Q = \left\{ q : q(x) = \prod_{j=1}^{j} q_j(x_j) \right\} \tag{3.9}$$

3.2.3 Bringing the pieces together

So far we have introduced two basic building blocks of model-based machine learning: graphical models and approximate inference. Together, they can be used to create highly tailored models of dynamical processes on networks. In a typical workflow, we start with the definition of a graphical model that encodes our beliefs about the dynamics generating data. We then couple the model with an inference procedure that learns about the dynamics from the data. As a result, we have a customised machine-learning model that can be used to understand the dynamics and predict their evolution in time (see Figure 3.1). The procedure can be repeated iteratively, building models and revising them in light of their predictive limitations [18], or just repeated for a few candidate models to compare their fitness to data using Bayesian Ockham's razor.

A flexible environment for developing model-based machine-learning applications is known as *probabilistic programming*. The basic idea of probabilistic programming is to provide a software tool for efficient inference in graphical models. It can be viewed as an extension of classical programming tailored to code with random variables, represented in terms of probability distributions. A user of the tool encodes a graphical model using probability distributions over model variables; the tool then automatically infers parameters of the model from data using a 'universal inference engine' [7]. The inference engine is usually implemented with Monte Carlo sampling techniques due to their generality to different problems. There is a growing number of probabilistic programming languages. Languages such as

BUGS [19], Stan [20] or Infer.NET [21] restrict the class of models that they can represent, but in return provide fast inference in comparison to more general languages such as Church [22], Venture [23] or Anglican [24]. There has been recently a lot of emphasis on developing fast inference in general models ([25] for example).

Probabilistic programming obviates the need to manually derive inference methods for graphical models and creates a very clear separation between the model and inference procedures. As such, it allows for rapid prototyping and testing of different models of data. This could make probabilistic programming revolutionary for both machine learning and scientific modelling.

3.3 Sequential data

So far in this chapter, we have introduced the general framework of model-based machine learning. The approach starts with a model that encodes our beliefs about the dynamics generating data. In our case study, as introduced in Section 2.3, the data are timestamped observations of vehicle positions. Intuitively, we expect that successive observations of positions are highly correlated. Furthermore, we expect that recent observations are likely to be more be informative than more historical observations in predicting future vehicle positions. All in all, we are faced with an example of sequential data.

3.3.1 Markov models

The easiest way to model sequential data would be to ignore the sequential nature and treat the observations as i.i.d. This approach, however, would fail to exploit the sequential patterns in data. Suppose, for instance, that we wish to predict vehicle position in a close time proximity. We know that the future position is going to be close to the current vehicle position. If we treat the data as i.i.d., then we can no longer use the temporal dependency to make future predictions.

To capture sequential patterns in a probabilistic model, one of the simplest ways is to consider a *Markov model*. First of all, note that, without loss of generality, we can use the product rule to express the joint distribution for a sequence of N observations $y_1, ..., y_N$ as

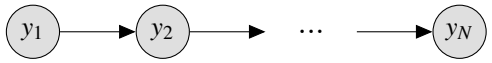


Figure 3.3: Graphical structure for a first-order Markov chain.

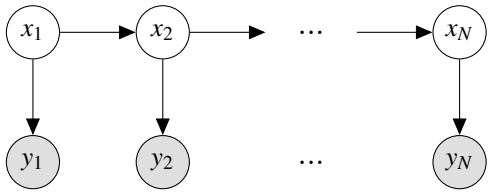


Figure 3.4: Graphical structure for the hidden Markov model and other state space Markov models.

$$p(y_1,...,y_N) = \prod_{n=2}^{N} p(y_n|y_1,...,y_{n-1}). \tag{3.10}$$

If we assume that current observation y_n is independent of all previous observations except for the most recent one y_{n-1}, we obtain the *first-order Markov chain* depicted as a graphical model in Figure 3.3. The joint distribution of the first-order Markov chain factorises as follows

$$p(y_1,...,y_N) = \prod_{n=2}^{N} p(y_n|y_{n-1}). \tag{3.11}$$

We can introduce additional latent variables to the model to allow additional modelling flexibility. For each observation y_n, we introduce a latent variable x_n. We now assume that the latent variables form a Markov chain, and that they give rise to the corresponding observed variables. The resulting graphical model is known as *state space Markov model* and is shown in Figure 3.4. The state space model corresponds to the joint distribution given by

$$p(y_1,...,y_N,x_1,...,x_N) = p(x_1)\prod_{n=1}^{N} p(x_n|x_{n-1})\prod_{n=1}^{N} p(y_n|x_n) \tag{3.12}$$

In most applications of state space models, the *transition* distribution $p(x_n|x_{n-1})$ and the *observation* distribution $p(y_n|x_n)$ that define the model are time-invariant, that is, they are independent of n. The model is then known as a *homogeneous* state space model.

3.3.1.1 Hidden Markov models

If latent variables x_n are discrete, then we deal with a special case of state space Markov models known as the hidden Markov Model (HMM) [26].

Since the latent variables are discrete, it is convenient to use a 1-of-K encoding scheme to turn them into K-dimensional binary variables, where K is the number of possible values they can take. As a result of the encoding, the conditional distribution $p(x_n|x_{n-1})$ corresponds to a matrix of numbers denoted by A, the elements of which are known as the *transition* probabilities. They are given by $A_{jk} = p(x_{nk} = 1|x_{n-1,j} = 1)$, and because they are probabilities, they satisfy $0 \leq A_{jk} \leq 1$ with $\sum_k A_{jk} = 1$. We can then write the conditional distribution explicitly in the form

$$p(x_n|x_{n-1,A}) = \prod_{k=1}^{K} \prod_{j=1}^{K} A_{jk}^{x_{n-1,j}, x_{n,k}} \tag{3.13}$$

The specification of the probabilistic model is completed by defining the observation probability $p(y_n|x_n)$. Because y_n is observed, the distribution $p(y_n|x_n)$ consists of a vector of K numbers corresponding to the K possible states of the binary vector x_n. If we denote by $\phi = \phi_1, ..., \phi_K$ the set of parameters governing the distribution, we can represent the observation probabilities in the form

$$p(y_n|x_n) = \prod_{k=1}^{K} p(y_n|\phi_k)^{x_{n,k}} \tag{3.14}$$

3.3.2 Exact inference

In many applications of interest, we might want to find the most probable sequence of latent variables that gave rise to the observed data. For example in robot localisation, we may wish to find a sequence of robot positions given sensor measurements. In the special case of hidden Markov models, where the latent variables are discrete, we can enumerate all possible sequences of latent variables to find the one that results in the highest joint probability over the latent states and the observations.

Viterbi algorithm [16] is a dynamic programming technique that efficiently searches the space of all possible latent sequences by recursively evaluating the

maximum joint probability for each state at each time step as follows

$$w_{nk} = p(y_n|x_{nk}) \cdot max_j[w_{n-1,j} \cdot p(x_{nk}|x_{n-1,j})] \tag{3.15}$$

with w_{1k} initialised to $w_{1,k} = p(y_1|x_{1,k})$. In the above equation, w_{nk} represents the joint probability of the most likely sequence of hidden states until time step n. Once the recursion reaches time step $n = N$, the most likely path is formed by the most likely sequence of hidden states and the shortest paths between them.

3.3.3 Particle filters

The general class of state space Markov models provide an extremely flexible framework for modelling sequential data. The great descriptive power of the models comes at the expense of intractability. It is generally impossible to obtain analytic solutions to inference problems with such models with the exception of a small number of particularly simple cases, such as hidden Markov models in Section 3.3.2. Particle filter algorithms described in this section are a broad and popular class of Sequential Monte Carlo algorithms which have been developed to approximate solutions to these intractable inference problems.

3.3.3.1 Sequential Monte Carlo

Consider a directed graphical model comprising of N latent variables x and N observed variables y, whose joint distribution factorises as

$$p(x_{1:N}, y_{1:N}) = p(x_1)p(y_1|x_1)\prod_{n=2}^{N} p(x_n|x_{1:n-1})p(y_n|x_{1:n}, y_{1:n-1}) \tag{3.16}$$

and, consequently, the *marginal likelihood*, $p(y_{1:N})$, is given by

$$p(y_{1:N}) = p(y_{1:N-1})p(y_N|y_{1:N-1}) \tag{3.17}$$

This form subsumes common state-space models, such as hidden Markov models (HMMs), as well as non-Markovian models, such as Gaussian processes.

The goal of sequential importance sampling is to approximate the posterior distribution $\pi(x) \equiv p(x_{1:N}|y_{1:N})$ by sampling from a (presumably simpler) proposal

distribution $q(x_{1:N}|y_{1:N})$ and then weighing the samples according to the *unnormalised weight* function

$$w(x_{1:N}) = \frac{p(x_{1:N}, y_{1:N})}{q(x_{1:N}|y_{1:N})} \tag{3.18}$$

Any form of proposal distribution can be used in principle, but a particularly convenient one takes the same factorisation as the true posterior

$$q(x_{1:N}|y_{1:N}) = q_1(x_1|y_1) \prod_{n=2}^{N} q_n(x_n|x_{1:n-1}, y_{1:n}). \tag{3.19}$$

The factorisations in (3.16) and (3.19) then lead to the recursive form of the *unnormalised* weight function

$$w(x_{1:N}) = w_1(x_1) \prod_{n=2}^{N} w_n(x_{1:n}) \tag{3.20}$$

where

$$\begin{aligned} w_1(x_1) &= \frac{p(x_1)p(y_1|x_1)}{q_1(x_1|y_1)} \\ w_n(x_{1:n}) &= \frac{p(x_n|x_{1:n-1})p(y_n|x_{1:n}, y_{1:n-1})}{q_n(x_n|x_{1:n-1}, y_{1:n})}. \end{aligned} \tag{3.21}$$

Sequential Monte Carlo methods approximate the target distribution $\pi(x_{1:N})$ sequentially by sampling from a sequence of intermediate distributions $\{\pi_n(x_{1:n})\}$ of increasing dimension $n = 1, ..., N$, where the final $\pi_N(x_{1:N}) \equiv \pi(x_{1:N})$ is the full target posterior of interest. That is, SMC first provide an approximation of $\pi_1(x_1)$, then an approximation of $\pi_2(x_{1:2})$ and so on. At each step n, SMC require an intermediate proposal distribution $q_n(x_n|x_{1:n-1}, y_{1:n})$ that approximates the transition from $x_{1:n-1}$ to x_n. The sequential approximation is known in the wider literature as the problem of *filtering*.

Procedurally, we initialize an SMC algorithm at $n = 1$ by sampling K values of x_1 from a proposal density $q_1(x_1|y_1)$ also known as the *initialisation probability* $p(x_1)$. We assign each of these particles x_1^k an importance weight $w_1(x_1^k)$ and then resample the particles according to their normalised weights W_1^k, where

$$W_1^k = \frac{w_1(x_1^k)}{\sum_{j=1}^K w_1(x_1^j)}. \tag{3.22}$$

The resampling step has the effect of removing particles with low weights and multiplying particles with high weights. We repeat the process for $n = 2, ..., N$, extending each particle $x_{1:n-1}^k$ by sampling a value x_n^k from a proposal kernel $q_n(x_n|x_{1:n-1}^k, y_{1:n})$, computing their unnormalised weights $w_n(x_{1:n}^k)$ and resampling according to the normalised weights W_n^k

$$W_n^k = \frac{w_n(x_{1:n}^k)}{\sum_{j=1}^K w_n(x_{1:n}^j)}. \tag{3.23}$$

At time n, the approximation of $p(y_n|y_{1:n-1})$ after sampling is

$$p(y_n|y_{1:n-1}) = \frac{1}{K}\sum_{k=1}^K w_n(x_{1:n}^k) \tag{3.24}$$

Hence an estimate of the marginal likelihood, by (3.17), is given by

$$\begin{aligned} p(y_{1:N}) &= \prod_{n=1}^N p(y_n|y_{1:n-1}) \\ &= \prod_{n=1}^N \frac{1}{K}\sum_{k=1}^K w_n(x_{1:n}^k) \end{aligned} \tag{3.25}$$

Once we reach $n = N$, our particles approximate samples from the target distribution $\pi(x_{1:N})$.

SMC methods perform best when the intermediate proposal distribution q_n is close to the optimal proposal q_n^{opt}, derived directly from the incremental change in densities [27]:

$$\begin{aligned} q_n^{opt}(x_n|x_{1:n-1}, y_{1:n}) &= \frac{p(x_{1:n}|y_{1:n})}{p(x_{1:n-1}|y_{1:n-1})} \\ &\propto \frac{p(x_{1:n}, y_{1:n})}{p(x_{1:n-1}, y_{1:n-1})} \\ &\propto p(x_n|x_{1:n-1})p(y_n|x_{1:n}, y_{1:n-1}). \end{aligned} \tag{3.26}$$

3.3.3.2 Bootstrap particle filters

In practice, the optimal proposal in (3.26) is almost always intractable, thus they are often approximated using a prior distribution, as in the case of bootstrap particle filter (PF) [28]:

$$q_n^{PF}(x_n|x_{1:n-1}, y_{1:n}) = p(x_n|x_{1:n-1}) \tag{3.27}$$

The corresponding importance weight in (3.21) then simplifies to

$$w_n^{PF}(x_{1:n}) = p(y_n|x_{1:n}, y_{1:n-1}). \tag{3.28}$$

The most common assumption of bootstrap particle filters is that your system satisfies *Markov properties* (depicted in Figure 3.4):

$$\begin{aligned} p(x_n|x_{1:n-1}) &= p(x_n|x_{n-1}) \\ p(y_n|x_{1:n}, y_{1:n-1}) &= p(y_n|x_n) \end{aligned} \tag{3.29}$$

This conveniently reduces (3.27) and (3.28) to *transition* and *observation* probabilities respectively:

$$\begin{aligned} q_n^{PF}(x_n|x_{1:n-1}, y_{1:n}) &= p(x_n|x_{n-1}) \\ w_n^{PF}(x_{1:n}) &= p(y_n|x_n). \end{aligned} \tag{3.30}$$

The most basic implementation of bootstrap particle filters is then given by the following algorithm:

Algorithm 1 Bootstrap particle filters for state space Markov models.

- **Initialisation**: At time $n = 1$, draw K particles according to initialisation probability $p(x_1)$. Call this set of particles X_1.
- **Recursion**: At time $n > 1$, generate a particle x_n for each particle in X_{n-1} by sampling from the transition probability $p(x_n \mid x_{n-1})$. Call the resulting set $\overline{X}_n$. Subsequently, draw K particles (with replacement) with a probability proportional to the observation probability $p(y_n \mid x_n)$. The resulting set of particles is X_n.

When the recursion reaches the last measurement at $n = N$, the particles stored in X_N are approximate samples from the desired distribution $p(x_{1:N} \mid y_{1:N})$.

Bootstrap particle filter is a popular approach to approximate inference in non-linear state space models and has appeared in the literature under various names including the bootstrap filter [28], survival of the fittest [29], and the condensation algorithm [30].

3.3.3.3 Limitations of particle filters

The algorithm described earlier suffers from several limitations. Firstly, in common with any importance sampling method, the performance of particle filters is strongly dependent on the choice of the proposal distribution. If the proposal is not well matched to the target distribution, then the method produces samples that have low effective sample size and, as a result, it requires a prohibitively large number of particles to represent the target distribution accurately. Secondly, it is important to emphasise that, even if the optimal importance distribution $p(x_n|y_n, x_{n-1})$ can be used, this does not guarantee that the SMC algorithms will be efficient. Indeed, if the variance of $p(y_n|x_{n-1})$ is high, then the variance of the resulting approximation with be high. Below we summarise some recent work that address the two shortcomings.

Building better proposals The particle filter community has developed various approaches to improve the quality of the proposal distribution. One approach attaches a post-sampling step that moves particles sampled from the proposal distribution towards the target distribution using Markov Chain Monte Carlo moves [31, 32, 33] or by solving partial differential equations [34, 35, 36, 37]. An alternative approach improves the proposal distribution by giving it additional information about the current [38, 39, 40] or even future observations [41] or their approximations [42]. Several authors considered conditioning the proposal distribution on the current observation only [43, 32]. The approaches successfully increased the effective sample size, but, often at the cost of high computational complexity or analytical intractability. The construction of good, but also computationally efficient proposal distributions is still an open research question.

Reducing variance Variance has been typically reduced using so-called Rao-Blackwellization, which constitutes the replacement of Monte Carlo sampling with analytical evaluation. Let us start by quoting Trotter [44]: "A good Monte Carlo is a dead Monte Carlo". Trotter specialised in Monte Carlo methods and did not advocate that we should not use them, but rather that we should avoid them whenever possible. In particular, whenever an integral can be computer analytically doing so should be preferred to the use of Monte Carlo techniques. The idea has given rise to a range of approaches that exploit the form of the system dynamics to partially (or fully) replace Monte Carlo sampling in particle filters with analytical computation [45, 46, 47, 48].

3.3.4 Performance evaluation

Models of sequential data can describe temporal dynamics and predict their evolution in time. They require validation metrics that are considerate of the temporal dependence of their predictions. In this section, we briefly review common approaches to validating machine learning models for time series data.

When developing a machine learning model, we typically split our data into a train and a test set: the training set used to prepare the model and the test set used to evaluate it. We may extend the idea to k-fold cross validation that repeats this process by randomly splitting the data into k groups, each given a chance to be a held out test set. These methods cannot be readily used with time series data as they assume no relationship between observations. Instead, we should split up the data in a way that respects the temporal order in which data points were observed. In the field of time series forecasting, the evaluation of models on historical data is called *backtesting*.

There are two basic approaches to backtest a machine learning model for time series problems:

- **Train-test split** respective of the temporal order of observations. This approach splits the data intro the train and test sets by selecting an arbitrary split point in the ordered list of observations and creating two new datasets. Depending on the amount of data available, one could consider split ratios

of 50-50, 70-30 and 90-10. The process can be repeated multiple times with different split points to provide a more robust estimate of the expected perfomance of the model on unseen data.

- **Walk-forward validation** that enables model evaluation as new observations arrive. In practice, we are often interested if the model can make good forecasts at each time step. Walk forward validation measures the performance of the model under this assumption. It consists of the following steps.

    1. First, decide on the number of observations used for model training. This may be thought of as the width of a sliding window over the available data.
    2. Start the sliding window at the beginning of the time series and use the samples in the window to train a model.
    3. The model makes a prediction for the next time step.
    4. The prediction is stored or evaluated against the known value.
    5. The window is moved by one data point to include the known value and the process is repeated.

    The approach creates many models, which might come at a high computationally intensive. This has, however, the benefit of providing a much more robust estimation of how the chosen modelling method and parameters will perform in practice.

Chapter 4

Where you are: inferring locations from noisy tracking data

Increasing availability of vehicle tracking data, in the form of GPS traces, has created potentially transformative opportunities for traffic management, route planning and other location-based services. Critical to the utility of the data is their accuracy. Map-matching is the process of improving the accuracy by aligning tracking data with the road network. In this chapter, we propose a range of approaches to map-matching based on Bayesian inference in state space hidden Markov models. We outline strengths and weaknesses of the proposed algorithms and compare their performance on tracking data of varied quality.

4.1 Why tracking data fails to locate you in cities

Location and navigation using global positioning systems (GPS) are deeply embedded in our daily lives. We use GPS to navigate between places and rely on GPS-based services for our daily needs, such as Uber for taxi services or Deliveroo for food deliveries. Despite our increasing dependence on GPS technologies, the fundamentals of how GPS works have not changed that much since its (literal!) launch in 1973, which leads to significant performance limitations. While GPS works well under clear skies, its location estimates can be wildly inaccurate (with an error of 50 meters or more) when we need it the most: in densely populated and highly built urban areas, where many of us are located.

In this section, we briefly discuss why GPS can perform poorly in urban environments before we outline how we fix it using novel map-matching algorithms in the sections that follow.

4.1.1 A short background on GPS

Let us begin with a quick recap of how GPS works in order to understand why it can be inaccurate in high-rise urban environments.

GPS is a network of more than 30 satellites operated by the U.S. government, orbiting the earth at an altitude of about 20,000 kilometres. These satellites send out radio frequency signals that GPS receivers, such as those found in smart phones, can lock onto. Importantly, the satellites record the time at which they launch their signal.

For each satellite whose signal the GPS receiver detects, the *pseudorange* is calculated, which is the difference between reception time and launch time (time-of-flight), multiplied by the speed of light. If the clocks of the satellite and the receiver were synchronised, the pseudorange would be equal to the straight line-of-sight distance to the satellite. However, the clocks are not synchronised, so the receiver needs to detect at least four satellites in order to calculate its own 3D coordinates on the globe and its clock bias (four unknowns). In practice, the receiver processes signals from up to 20 GPS satellites, and having more than the required minimum of four satellites improves the robustness of its location estimation against noise or blockages.

4.1.2 Why GPS fails in urban environments

A major assumption behind GPS-based positioning is that the receiver has an unobstructed line of sight to each satellite whose pseudorange it is computing. This assumption holds in open terrain, but breaks down in dense urban environments, as shown in Figure 4.1.

Buildings often block the lines of sight to satellites, so the receiver frequently processes signals corresponding to reflections of other buildings. The inaccuracy in pseudoranges resulting from this phenomenon can cause errors in location esti-

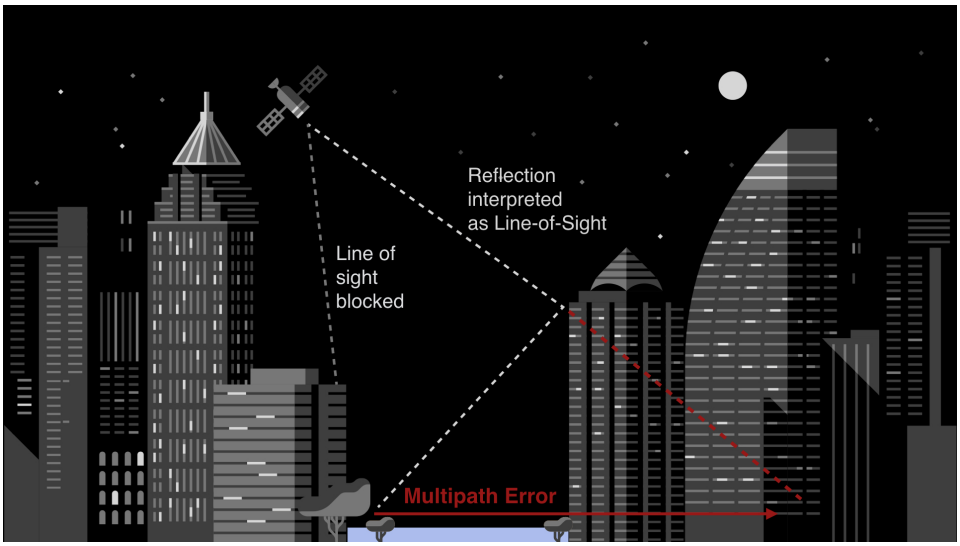


Figure 4.1: Line-of-sight blockage and strong reflections can cause large GPS errors. Source: Uber Engineering blog (https://eng.uber.com/).

mates of 50 meters or more in urban canyons. Most of us have suffered from this phenomenon first-hand when using GPS navigation in urban areas.

4.2 Overview of our approach

Over the last years we have witnessed a rapid increase in the availability of GPS-receiving devices, such as smart phones or car navigation systems. The devices generate vast amounts of temporal positioning data that have been proven invaluable in various applications, from traffic management [49] and route planning [50, 51, 52] to inferring personal movement signatures [53].

Critical to the utility of GPS data is their accuracy. As mentioned in Section 4.1, the data suffer from measurement errors caused by the technical limitations of GPS receivers and sampling errors caused by their receiving rates. The data are particularly erroneous where we need them the most: in urban environments, where buildings reflect satellite signals causing large measurements errors. The accuracy of the data can be improved by averaging multiple trajectories [54, 55]. Alternatively, when digital maps are available, it is common practice to improve the accuracy of the data by aligning GPS points with the road network. The process is known as *map-matching*.

Most map-matching approaches use *positions* of GPS points in order to align

them with the street network (see Figure 4.2). The approaches could be broadly categorised into *local* and *global*. Local approaches typically employ iterative schemes, in which a new GPS point is matched to the street network by considering its position and how preceding positions were matched, whereas global approaches align entire GPS trajectories at once. Local algorithms tend to be computationally cheaper but less accurate than their global alternatives. Both approaches completely overlook the temporal dimension of GPS trajectories, making them inefficient in cases when the sampling rate is low or the street network complexity is high as then speed information is useful in distinguishing between alternative roads.

More advanced map-matching techniques utilise both *timestamps* and *positions* of GPS points in order to achieve a higher degree of accuracy. A popular example of a spatio-temporal algorithm is ST-matching [2]. It is easy to implement and shows improved accuracy over purely spatial approaches, especially when GPS sampling frequency is low. The major limitation of the technique and the beforementioned spatial approaches is their deterministic nature. They would always snap a GPS trajectory to a road network, regardless if it even came from the road network in the first place! The lack of confidence scores associated with their outputs might lead to very misleading results, especially when the data quality is low.

This chapter addresses the issue of certainty by proposing purely probabilistic spatio-temporal map-matching approaches. The approaches are examples of model-based machine learning. They use a state space hidden Markov model (see Figure 3.4) to represent the relationship between true positions (unobserved) and the observed GPS measurements. Then, they perform map-matching as Bayesian inference in the Markov model, inferring the unobserved true positions from noisy GPS data. The three proposed models are:

- **PST-Matching**: a probabilistic adaptation of ST-Matching, based on Viterbi inference in hidden Markov model;

- **particle filters**: map-matching based on approximate inference in state space hidden Markov model using particle filters;

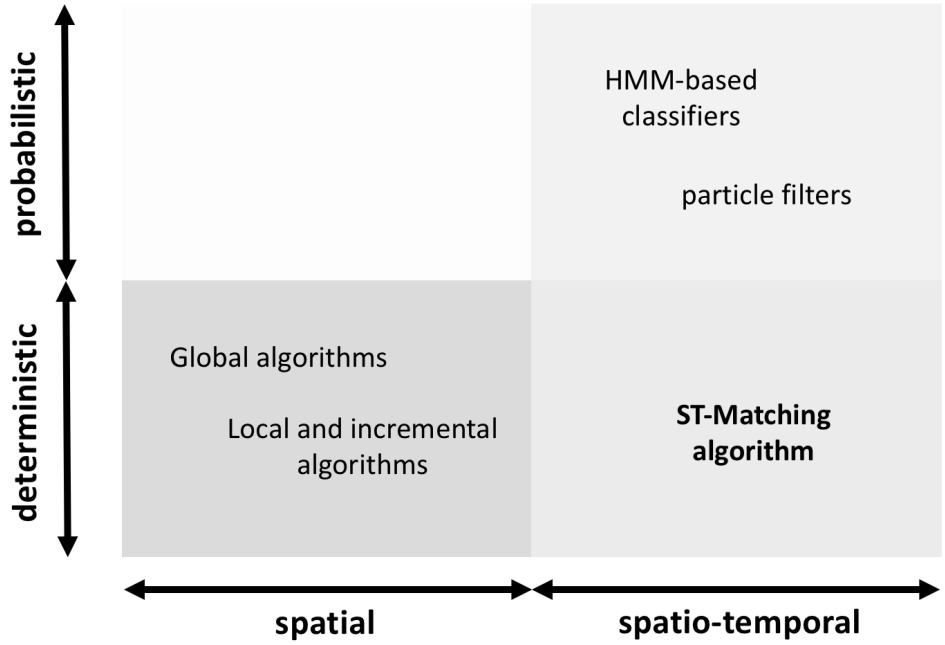


Figure 4.2: A categorisation of map-matching algorithms.

- **look-ahead particle filters**: improved map-matching for sparse GPS data based on an adaptation of particle filters.

Our approaches could be divided into two types: the first one assumes there are only a few possible positions for each GPS measurement (PST-Matching), the second one relaxes that assumption to a distribution over all possible positions on the road network (particle filters, look-ahead particle filters). The first approach corresponds to an inference problem with a discrete latent variable, which can be solved exactly using Viterbi algorithm (see Section 3.3.2). The second one allows for extra flexibility at the expense of intractability: the inference problem can only be approximated using particle filters (see Section 3.3.3). Both types of algorithms use spatial and temporal information to align a GPS trajectory with the road network. They output the most likely road sequence that the GPS data came from together with the associated likelihood, which places them in the upper right quadrant of the map-matching classification in Figure 4.2.

The remainder of the chapter is outlined as follows. It begins by formalising the problem of map-matching in Section 4.3. It then outlines method validation in Section 4.4. Finally, it introduces the three proposed map-matching methods. It

thoroughly evaluates the performance of each method using a range of GPS trajectories of varied frequencies and levels of noise.

4.3 Problem statement

In this section, we define the problem of probabilistic map-matching.

Definition 4.3.1 *(GPS trajectory) GPS trajectory is a sequence of GPS points, where each GPS point contains latitude, longitude, bearing and timestamp.*

Definition 4.3.2 *(Road network) Road network is a directed graph with vertices representing road intersections and edges representing road segments. Bidirectional road segments are represented by two edges, each corresponding to a single direction of flow. Roads and intersections can be uniquely identified using their IDs.*

Definition 4.3.3 *(Path) Path is a connected sequence of street segments in the road network.*

Given a road network and a GPS trajectory, the goal of probabilistic map-matching is 1) to find the most likely path that the GPS trajectory was generated from and 2) to quantify the confidence that the path is indeed the true path taken.

Figure 4.3: Example road network with a GPS trajectory to be map-matched.

4.4 Method validation

Data: The dataset used for validating the proposed algorithms is a complete GPS trajectory of a police patrol vehicle during its night shift (9pm to 7am) in the London Borough of Camden on March 9th 2011. The dataset contains 4,800 GPS points that were emitted roughly every second when moving. It is part of a larger database of police tracking data, introduced in Section 2.3, which has motivated the research presented in this thesis.

Further datasets of degraded quality are artificially created from the acquired data in order to test the robustness of the proposed algorithm on a range of GPS trajectories of varied sampling rates and levels of noise. Their sampling rate is manipulated by removing GPS points at chosen intervals. Their level of noise is controlled by perturbing GPS point by Gaussian noise with zero mean and a chosen standard deviation. There is already some random Gaussian noise inherent to the data. However, since Gaussian distributions are additive, i.e. adding two Gaussian random variables results in another Gaussian random variable with mean and variance equal to the sum of the added means and variances, any additional amount of noise can be simulated once the standard deviation of the original noise distribution is empirically found.

Accuracy testing: Since there is no ground truth available, we follow the technique of walk-forward validation introduced in Section 3.3.4. The technique is designed for online forecasting models, so it requires some adaptation to map-matching methods which work predominantly offline. Instead of leaving out one future data point for method validation, we leave out one data point in between sequences of training points. We split available GPS sequences according to the split ratio of 9:1, nine training data points followed by one test data point. In practice, this equates to us removing every 10th GPS point from each GPS trajectory for training (see Figure 4.4). We proceed by aligning the trajectory of training GPS points with the road network using our proposed PST-Matching algorithm. We then record how far off the predicted path each test point is. The more off, the more erroneous our map-matching proposal. We use the average distance across all test points as the

measure of map-matching error made.

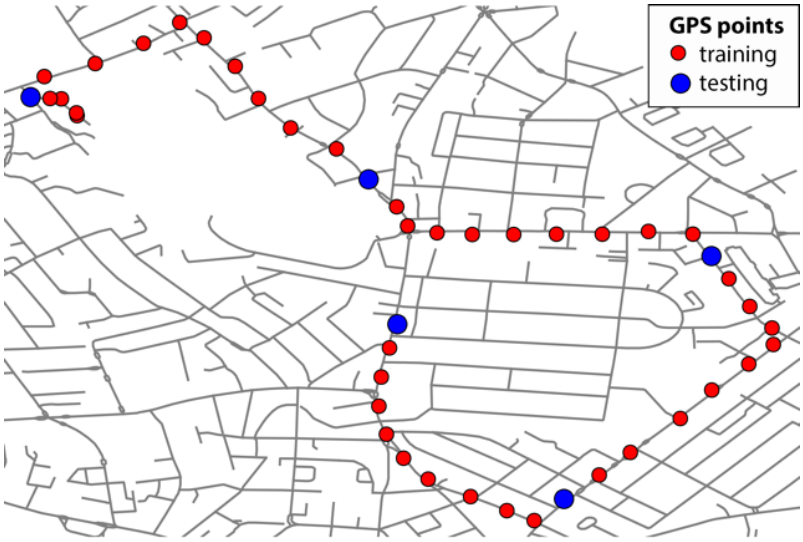


Figure 4.4: Example GPS trajectory with points split into training and test sets.

4.5 Proposed methods

4.5.1 Probabilistic ST-Matching

4.5.1.1 Introduction

The ability to infer routes taken by vehicles from sparse and noisy GPS data is of crucial importance in many traffic applications. The task, known as map-matching, can be accurately approached by a popular technique known as ST-Matching. The technique considers both *timestamps* and *positions* of GPS points in order to achieve a high degree of map-matching accuracy. It uses spatial information to find candidate roads for each GPS point and then seeks a sequence of candidate roads that best matches the temporal profile of the GPS trajectory. The algorithm is easy to implement, computationally efficient and has been shown to outperform purely spatial map-matching approaches, especially when the sampling rate is low. The algorithm can align any given trajectory with the road network, even if the trajectory is very noisy or erroneous. Unfortunately, it is deterministic in nature, which means that it does not indicate the confidence that the alignment is correct. The determinism might lead to very misleading results, especially when the data quality is low.

In isolation from deterministic algorithms, such as ST-Matching, *probabilistic* approaches to map-matching have been designed that address the issue of confidence using probabilities. They also belong to the class of spatio-temporal techniques as they use both spatial and temporal information when calculating probabilities of specific map-matching outputs. They typically represent the map-matching problem using a hidden Markov Model (HMM) where hidden states are true positions that are learnt from noisy GPS trajectories [57, 58, 59]. The most likely map-matching output can then be efficiently learnt by applying a dynamic programming algorithm, such as the Viterbi algorithm, to the HMM lattice. Probabilistic approaches calculate the most likely or a few most likely road paths and output them together with their likelihoods. They are methodologically powerful, but largely isolated from the rest of the map-matching community, often limiting their uptake by researchers and practitioners from other fields.

In this section, we present a map-matching algorithm that bridges the gap between the deterministic and probabilistic classes of spatio-temporal algorithms. It is an adaptation of the well-established ST-Matching that turns it from being deterministic to fully probabilistic. The adaptation brings the best of the deterministic and probabilistic worlds into a highly accurate and computationally efficient map-matching algorithm that is capable of expressing levels of map-matching confidence. The proposal is inspired by apparent similarities between ST-Matching and HMM-based approaches to map-matching.

The section is outlined as follows. It begins by introducing ST-Matching and a general HMM-based framework as its probabilistic counterpart. It analyses similarities and differences between the two approaches in order to propose a probabilistic adaptation of ST-Matching, called probabilistic ST-Matching or PST-Matching in short. It evaluates the robustness of PST-Matching on a range of GPS trajectories of varied frequencies and levels of noise. Similarly to ST-Matching, the proposed algorithm shows high accuracy on datasets with low GPS frequency, yet with the added benefit of confidence scores associated with its outputs.

4.5.1.2 Methodology

In this section, we describe our probabilistic ST-Matching algorithm in detail. We begin by introducing its components: the ST-Matching algorithm (deterministic) and a general HMM-based approach (probabilistic). We then outline modifications required to make the ST-Matching algorithm fit the general HMM-based framework, thus turning it into a probabilistic technique.

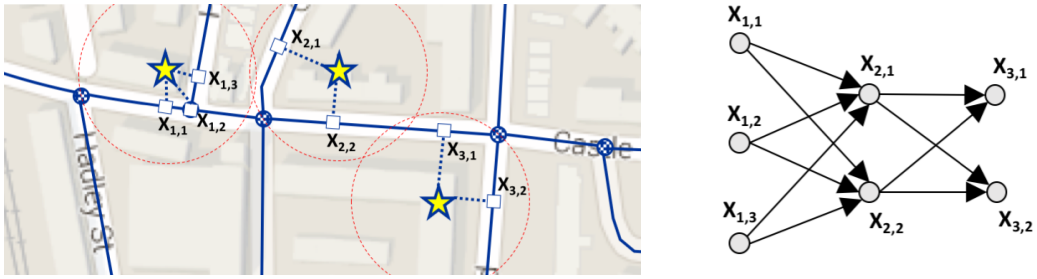
ST-Matching algorithm: A deterministic map-matching approach proposed by [2] that combines spatial and temporal information to effectively align low-sampling-rate GPS trajectories with the road network. It is easy to implement and has been shown to outperform more traditional map-matching approaches in terms of accuracy and running time. Its architecture consists of two basic steps: candidate graph preparation and best path computation.

- *Candidate graph*:

The candidate graph stores all possible true paths given a GPS trajectory. Nodes of the graph are candidate position for each GPS observation, edges are shortest road paths between neighbouring candidate positions. The preparation of the graph involves the following steps.

Firstly, candidate positions are computed by retrieving road segments within radius r of each GPS observation and then finding a position on each segment at the shortest distance to the relevant observation. The positions can be anywhere on the underlying road network, such as along road segments or on road junctions. The procedure is exemplified in Figure 4.5. The obtained candidate positions are represented as nodes in the candidate graph. The number of candidate positions can differ among GPS observations, depending on the number of street segments within the search radius. In the section, we use $x_{i,j}$ to denote the j^{th} candidate position of GPS observation y_i.

Secondly, the shortest paths between pairs of candidate positions at adjacent time steps are evaluated based the road topology. They are represented as edges in the candidate graph, as shown in Figure 4.5b.



(a) candidate positions **(b)** candidate graph

Figure 4.5: Candidate graph generation for an example GPS trajectory with three observations (depicted as yellow stars). (a) Candidate positions x_i for each GPS observation $i = 1,...,3$ are computed as the closest points on each street segment within a fixed search radius around the observation. (b) The candidate positions are represented as nodes in the candidate graph. They are connected by edges if they correspond to adjacent GPS readings.

Finally, the nodes and edges of the candidate graph are weighted based on the spatio-temporal profile of the GPS trajectory and the topology of the underlying road network. Their weights reflect their *observation* and *transmission probabilities*, respectively.

Definition 4.5.1 *(Observation probability) Given a GPS observation y_i at time step i and a corresponding candidate position $x_{i,j}$, observation probability defines the probability that the GPS observation y_i is emitted from the candidate position $x_{i,j}$.*

The observation probability is specified as a Gaussian distribution of the distance between y_i and $x_{i,j}$:

$$\mathcal{N}(x_{i,j}) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp - \frac{(d_{i,j} - \mu)^2}{2\sigma^2} \tag{4.1}$$

where $d_{i,j}$ is the distance between y_i and $x_{i,j}$. The mean μ of the distribution is set to zero, the standard deviation σ is empirically estimated from the distances between GPS observations and their closest street segments. The empirical estimation validates that the observation noise is approximately Gaussian.

Definition 4.5.2 *(Transmission probability) Given two candidate positions $x_{i-1,j}$ and $x_{i,k}$ for two neighbouring GPS observations y_{i-1} and y_i, transmission probability defines the probability that the "true" path from y_{i-1} to y_i follows the shortest path from $x_{i-1,j}$ to $x_{i,k}$.*

As in the original paper [2] , the transmission probability is defined as follows:

$$V(x_{i-1,j} \rightarrow x_{i,k}) = \frac{e_{i-1 \rightarrow i}}{s_{(i-1,j) \rightarrow (i,k)}} \tag{4.2}$$

where $e_{i-1 \rightarrow i}$ is the Euclidean distance from y_{i-1} to y_i and $s_{(i-1,j) \rightarrow (i,k)}$ is the length of the shortest path from $x_{i-1,j}$ to $x_{i,k}$. The definition encodes our preference to follow the shortest path along the road network which cannot be shorter than the Euclidean distance to our destination.

The above definition only considers spatial information when calculating the likelihood of transmission. A spatio-temporal version of the transmission probability is also considered in the original paper [2] and could be easily incorporated into the methodology presented in this thesis in the future.

- *Best path search*:

Once the candidate graph is defined and its nodes and edges are weighted according to the observation and transmission probabilities, respectively, a dynamic programming method is applied in order to find a path through the graph with the maximum weight. The path represents the most likely "true" path that the GPS trajectory was generated from.

The method proposed in [2] calculates the most likely path by recursively evaluating the following equation:

$$f(n,k) = \mathcal{N}(x_{n,k}) + \max_j [f(n-1,j) \cdot V(c_{n-1,j} \rightarrow x_{n,k})] \tag{4.3}$$

with $f(1,k)$ initialised to $f(1,k) = \mathcal{N}(x_1,k)$. In the above equation, $f(n,k)$ represents the total weight of the most likely sequence of positions ending at position $x_{n,k}$,

based on GPS observations at time steps $1:n$. Once the recursion reaches $n = N$, the obtained sequence of positions and the shortest paths between them form the most likely path given the GPS trajectory.

General HMM-Based Approach: Hidden Markov Model (HMM) is an established framework for probabilistic time-series modelling (see Section 3.3.1.1). It provides a principled way of representing uncertainty in measurements taken over time and as such is suitable for modelling uncertainty inherent to noisy GPS trajectories.

There have been numerous probabilistic approaches to map-matching based on HMMs [57, 58, 59]. They typically use an HMM to represent possible "true" paths and their probabilities and then search for one or more paths with the highest probabilities as the map-matching output.

Drawing similarities to the ST-Matching algorithm, an HMM can be understood as a candidate graph from which the most likely path can be retrieved via a dynamic programming routine known as the Viterbi algorithm (introduced in Section 3.3.2).

- *Candidate graph*:

HMM provides a graph structure for storing possible paths in a probabilistic manner. Nodes of the graph are hidden states that can represent candidate positions at each time step. Edges are transitions between the hidden states and can represent possible paths taken between candidate positions at adjacent time steps. The structure of the graph in the context of map-matching is, in fact, equivalent to that of the candidate graph in ST-Matching (see Figure 4.5b for an example).

Nodes are assigned observation probabilities that quantify the likelihood of the observations given the hidden states at each time step. The observation probability is defined as the conditional probability $p(y_i|x_{i,j})$ of observing y_i given that the true state at time step i is $x_{i,j}$. In the map-matching context, it is equivalent to the observation probability given in Definition 4.5.1.

Edges are given so-called transition probabilities. The transition probability is

a discrete conditional distribution $p(x_{i,k}|x_{i-1,j})$ that defines the probability of transitioning from hidden state $x_{i-1,j}$ at time step $i-1$ to another hidden state $x_{i,k}$ at time step i. In our context, it is the probability of following the shortest path between candidate positions corresponding to these hidden states. The transition probability can be any discrete distribution, such as the ST-Matching transmission probability in Definition 4.5.2, but more rigorously defined to ensure that basic rules of probability are satisfied. In particular, the following statement must hold:

$$\sum_{k} p(x_{i,k}|x_{i-1,j}) = 1 \tag{4.4}$$

- *Best path search*:

The most likely path is inferred as the most likely sequence of hidden states using a dynamic programming technique known as the Viterbi algorithm [16], introduced in Section 3.3.2. The algorithm is an exact inference method that outputs the most likely path as the most likely sequence of hidden states and the shortest paths between them.

The Viterbi algorithm in (3.15) is almost equivalent to the ST-Matching algorithm in (4.3). If one replaced the observation and transmission probabilities in (4.3) with the more general observation and transition probabilities of the Viterbi algorithm, respectively, the ST-Matching algorithm would only differ in the way it applies the most recent observation probability to the result of the max operation (addition instead of multiplication). However, it lacks the probabilistic treatment of the Viterbi approach which not only finds the most likely path but also quantifies the likelihood that it is indeed the true path taken using its joint probability.

Probabilistic ST-Matching Algorithm: Having introduced the ST-Matching algorithm and a general HMM-based approach, it has become apparent that the two approaches share a lot of similarities. In this section, we formalise the observation and outline modifications required to make the ST-Matching algorithm fit the probabilistic framework, thus giving it the ability to express map-matching confidence

in a probabilistic manner. We term the proposed modification the probabilistic ST-Matching (PST-Matching) algorithm.

- *Candidate graph*:

Candidate graph of PST-Matching is very similar to the original ST-Matching graph. It shares the same graphical structure and defines the emission probability according to the same formula in (4.1). It requires a modified transmission probability, however, as the original definition in (4.2) does not satisfy basic rules of conditional probabilities, such as the summation rule in (4.4). We satisfy the requirement by proposing a normalised transmission probability V_{pst} (subscript *pst* stands for PST-Matching):

$$V_{pst}(x_{i-1,j} \to x_{i,k}) = \frac{V(x_{i-1,j} \to x_{i,k})}{\sum_k V(x_{i-1,j} \to x_{i,k})} \tag{4.5}$$

The normalisation makes the transmission probability robust against inconsistencies in temporal gaps between measurements. It also gives the transmission a clear probabilistic interpretation that the original definition in (4.2) is lacking.

- *Best path search*:

The dynamic programming routine of PST-Matching is a modification of that of ST-Matching (4.3) that turns it into a Viterbi algorithm. The modification simply requires replacing the addition operation in (4.3) with multiplication. When applied to the candidate graph outlined above, the proposed algorithm takes the following recursive form:

$$f_{pst}(n,k) = \mathcal{N}(x_{n,k}) \cdot \max_j [f_{pst}(n-1,j) \cdot V_{pst}(x_{n-1,j} \to x_{n,k})] \tag{4.6}$$

with $f_{pst}(1,k)$ initialised to $f_{pst}(1,k) = \mathcal{N}(x_{1,k})$. As in any Viterbi algorithm, the quantity stored in $f_{pst}(x,k)$ at the final time step is the joint probability of the most likely path. It serves as a measure of map-matching confidence that the original ST-Matching algorithm is lacking. Since the measure depends linearly on the

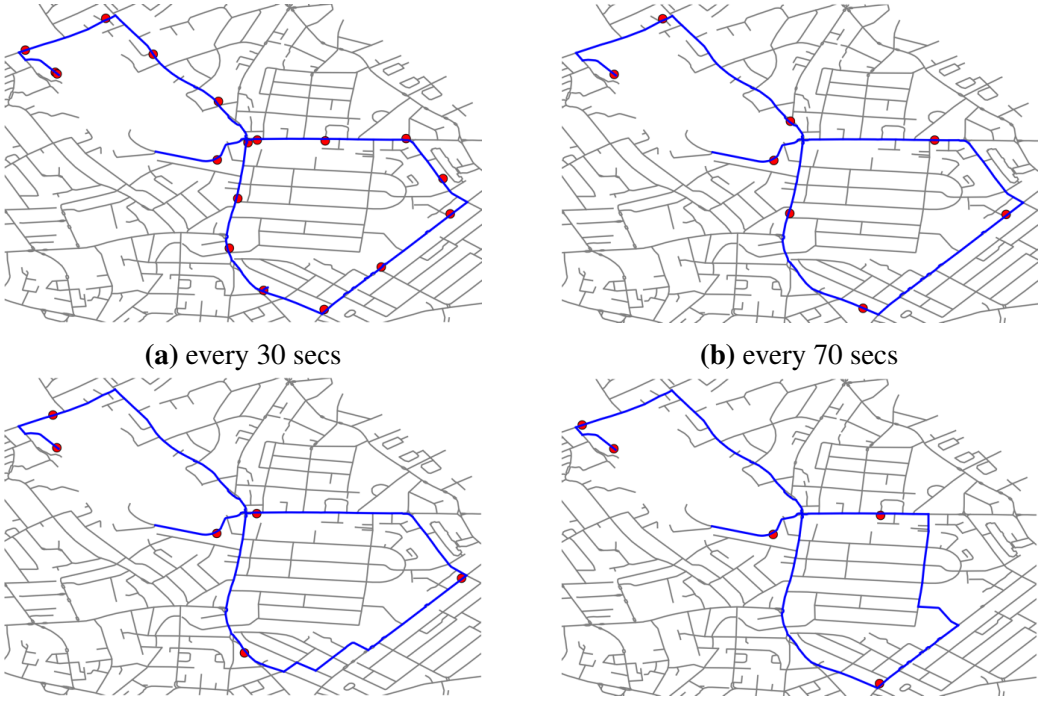
length of the input GPS sequence, we evaluate it on a sliding window of *fixed* length over the GPS sequence.

4.5.1.3 Results

We tested the proposed PST-Matching algorithm on datasets of varied quality and compared its accuracy against that of the original ST-Matching algorithm. See Figure 4.6 for example map-matching outputs and Figure 4.7 for a summary of the algorithm's performance across all datasets. Similarly to ST-Matching, the algorithm shows high accuracy on datasets with noise as high as 30 meters standard deviation and sampling rates of up to 90 seconds. Such extreme conditions are rarely found in real datasets; hence the algorithm should be successful on real GPS trajectories without the need for any prior adjustments or parameter fitting.

We noticed a slight drop in performance at very high sampling rates of 1-2 seconds (see Figure 4.7a). This is likely caused by the fact that frequent, noisy observations tend to pull rather violently towards different path proposals. The algorithm also gradually deteriorates at higher levels of measurement noise, unlike the original ST-Matching algorithm (see Figure 4.7b). This is due to the normalisation of the transmission probability according to (4.5), which reduces the variance of transmission probabilities. As a result, higher probabilities are assigned to candidate points that are clearly off the "true" path. As the measurement noise increases, there are more points off the path that PST-Matching accidentally includes in the most likely path.

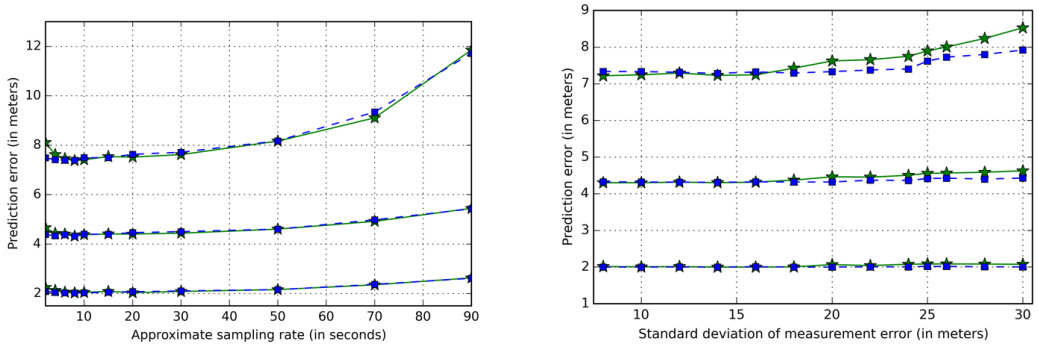
We investigated how confidence of PST-matching solutions, expressed as joint probabilities, changes with the sampling rate and the level of noise of GPS data. Since the joint probability of a solution is the product of observation and transmission probabilities along the most likely sequence (see (4.6)), its value depends on the length of the input GPS sequence. The dependence is linear as shown in Figure 4.8. We ensured that the dependence did not skew our analysis by applying PST-Matching to a sliding window (of length ten) over input GPS trajectories. The idea guaranteed that confidence scores were comparable and gave the algorithm the ability to process GPS trajectories in an online manner. The obtained confidence



(a) every 30 secs **(b)** every 70 secs

(c) every 110 secs **(d)** every 150 secs

Figure 4.6: Performance of PST-Matching at different sampling periods.



(a) sampling rate **(b)** measurement noise

Figure 4.7: Accuracy of PST-Matching (green) and ST-Matching (blue) on datasets with varied GPS sampling rates and noise represented as 25th, 50th and 75th percentiles of map-matching errors.

scores are shown in Figure 4.9. On average, the scores decline as data become noisy and sparse. This trend is exemplified in Figure 4.10, where after adding noise to the data, the quality and confidence of the map-matching output gradually drops. These intuitive results show that the confidence scores are closely aligned with the quality of map-matching results and, as such, could prove indispensable when dealing with GPS data of unknown quality.

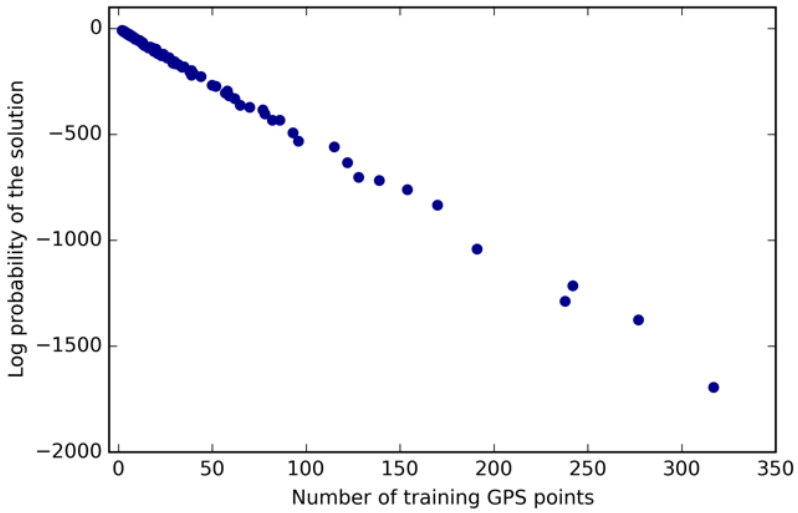
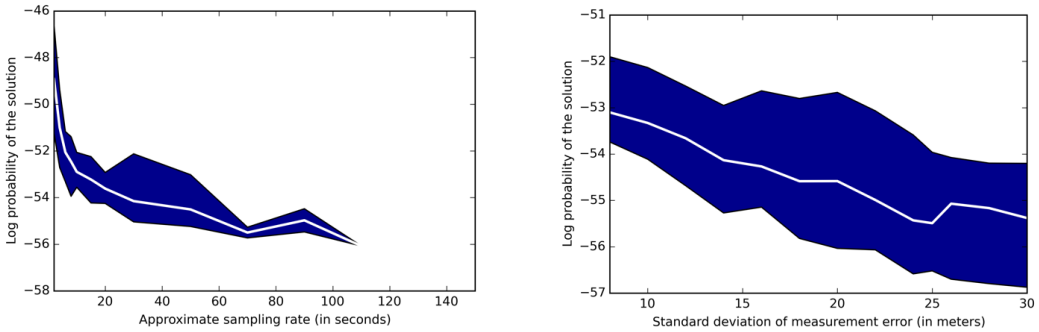


Figure 4.8: PST-Matching confidence as a function of journey length.

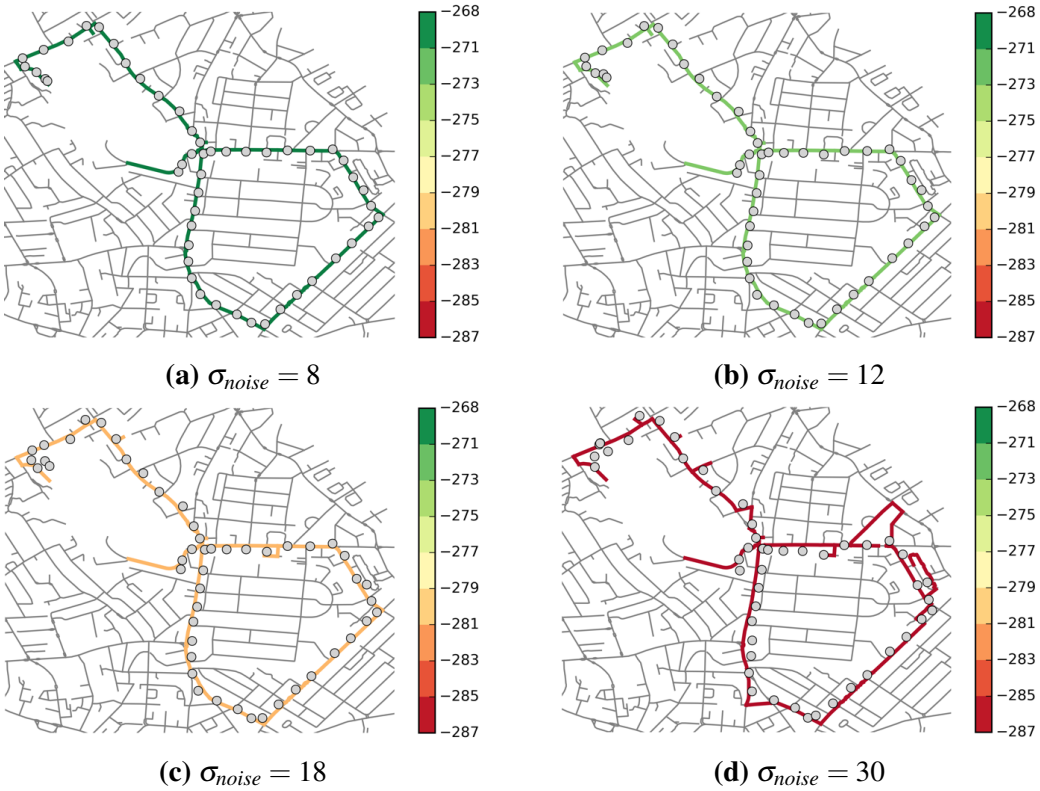


(a) sampling rate **(b)** measurement noise

Figure 4.9: Map-matching confidence, measured as the log of probabilities in (4.6), on datasets with varied GPS sampling rates and noise represented as 25th, 50th and 75th percentiles of log probabilities of map-matching outcomes.

4.5.1.4 Conclusions

We propose a new probabilistic map-matching algorithm called PST-Matching for aligning sparse and noisy GPS trajectories with a road network. The algorithm is a probabilistic extension of a popular deterministic algorithm called ST-Matching that has been shown to outperform more traditional map-matching algorithms on datasets of low sampling rates. The proposal brings high computational efficiency and accuracy of ST-Matching into the probabilistic world, hence giving it the ability to express confidence about its outputs. The measure of confidence is particularly important when dealing with traffic datasets of low accuracy. We validate the pro-



(a) $\sigma_{noise} = 8$ **(b)** $\sigma_{noise} = 12$

(c) $\sigma_{noise} = 18$ **(d)** $\sigma_{noise} = 30$

Figure 4.10: Example PST-Matching outcome with confidence expressed as log probabilities on a GPS trajectory with varied noise standard deviation (in meters).

posed algorithm on a range of GPS trajectories of varied quality to show that it has as high accuracy on low-frequency and noisy datasets as the original ST-Matching algorithm, yet with the added benefit of expressing beliefs about the quality of its output using probabilities.

4.5.2 Particle filters

4.5.2.1 Introduction

In recent years, increasing digitisation of vehicle traces has enabled new insights into urban dynamics. Critical to the utility of the data is their accuracy, which can be improved through the process of map-matching. In this chapter, we propose a purely probabilistic approach to map-matching based on a sequential Monte Carlo algorithm known as particle filters. The algorithm originates from the field of robotics [60], where it has been widely applied in robot localisation problems. In the context of map-matching, it uses both spatial and temporal information to iter-

atively align a GPS trajectory with the road network. In contrast to ST-Matching, it does not constrain the candidate positions for each GPS point to any pre-defined locations, but rather samples them from a distribution over the entire road network. It can also output multiple candidate solution, each with an associated probability score. The flexibility of particle filters comes at the expense of higher computational cost and, as we discover during this research project, higher sensitivity to low GPS sampling rate.

4.5.2.2 Methodology

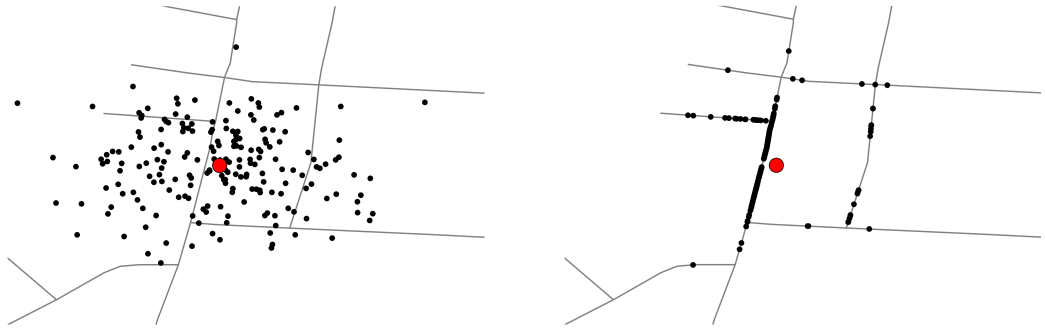
Our map-matching framework is based on particle filters. The algorithm computes candidate paths and their probabilistic values given a GPS trajectory. The most probable candidate path can then be selected as the map-matching outcome. The framework is evaluated using cross-validation.

Particle Filter: Bootstrap particle filter, previously introduced in Section 3.3.3, is a sequential Monte Carlo technique that approximately infers true states of a dynamical system given its noisy observations. In our case, the dynamical system is a vehicle following a path along the road network, noisy observations are GPS points and the true states that we want to infer are actual locations of the vehicle at different timestamps.

The algorithm is based on the assumption that the dynamical system can be modelled as a first-order Markov chain with unobserved (hidden) states shown in Figure 3.4. That is, it assumes that the state of the system x_n at time n solely depends on the state at time $n-1$ through the so-called *transition probability* $p(x_n|x_{n-1})$. It adds that any measurements of the system are noisy descriptions of the unobserved true states, where the noise is modelled by the *measurement probability* $p(y_n|x_n)$.

The goal of particle filters is to approximate samples from the posterior distribution $p(x_{1:n} \mid y_{1:n})$ over possible paths $x_{1:n}$ given all available measurements $y_{1:n}$. The algorithm approximates the solution by recursively sampling from the proposal distribution in (3.19), see Algorithm 1. The samples are represented by *particles*, i.e. possible states of the system given measurements.

Definition 4.5.3 *(Particle) A point on the road network containing unique road*



(a) unconstrained **(b)** constrained to the road network

Figure 4.11: Initialisation of particles around the first GPS point in a trajectory.

segment identifier, distance along the segment and direction of travel (defined by from-to endpoints of the segment).

Once the recursion in Algorithm 1 reaches the last measurement at $n = N$, the particles approximate samples from the desired distribution $p(x_{1:N} \mid y_{1:N})$. In our context, they represent possible paths taken by a vehicle given the GPS trajectory. The certainty associated with each path is proportional to the fraction of particles that it is represented by.

4.5.2.3 Results

Implementation: We follow the basic implementation of bootstrap particle filters outlined in Algorithm 1. The implementation requires specifying the initialisation, transition and observation probabilities that describe the dynamics of a vehicle following a path along the road network.

A **Initialisation**

The initialisation probability distribution $p(x_1)$ is defined as a Gaussian centred at the position and bearing of the first GPS point. Particles initialised from the distribution are required to be positioned on the road network, hence their positions are first sampled (see Figure 4.11a) and then either kept or discarded depending on whether they coincide with the road network or not (see Figure 4.11b). Their direction of travel is inferred from the sampled bearing.

B **Transition probability**

The transition probability $p(x_n \mid x_{n-1})$ is set as a linear estimate equal to the Cartesian distance between GPS points x_{n-1} and x_n plus an additive Gaussian noise.

In the recursive step of particle filter, particles move along the road network by a distance sampled from $p(x_n \mid x_{n-1})$. When they encounter a road intersection, they randomly choose which road to follow.

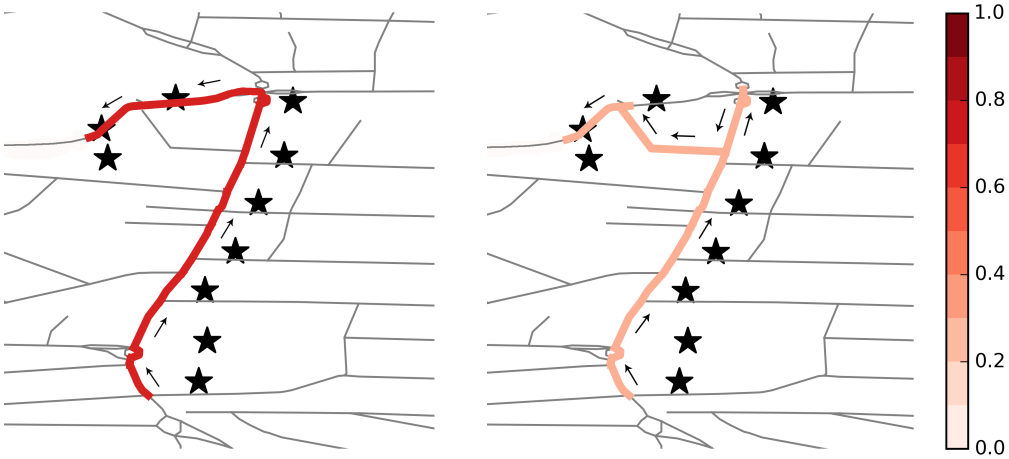
C **Measurement probability**

The measurement noise $p(y_n \mid x_n)$ is also modelled as a Gaussian distribution, i.e. it is expected that GPS points are normally distributed around the true vehicle locations.

D **Particle resampling**

Finally, the implementation follows the most basic resampling strategy which resamples particles at each time step. As we have previously mentioned, resampling has the effect of removing particles with low weights and multiplying particles with high weights. This is at the cost of immediately introducing some additional variance. Although resampling is not the focus of this research, in practice one could optimise the resampling strategy by looking at the variability of the particle weights using the so-called Effective Sample Size (ESS) criterion [61], which is later defined in (6.12).

Performance Evaluation: We follow the evaluation framework outlined in Section 4.4 to measure the accuracy of the algorithm on a range of GPS trajectories. In the first instance, the proposed algorithm is applied to the police vehicle data. An example output of the algorithm is shown in Figure 4.12. The median cross-validation error is 4.9 meters, i.e. the inferred paths tend to be 4.9 meters away from GPS points not included in the map-matching. The error approximately equals the measurement noise of the GPS data themselves, therefore the results seem to be accurate.



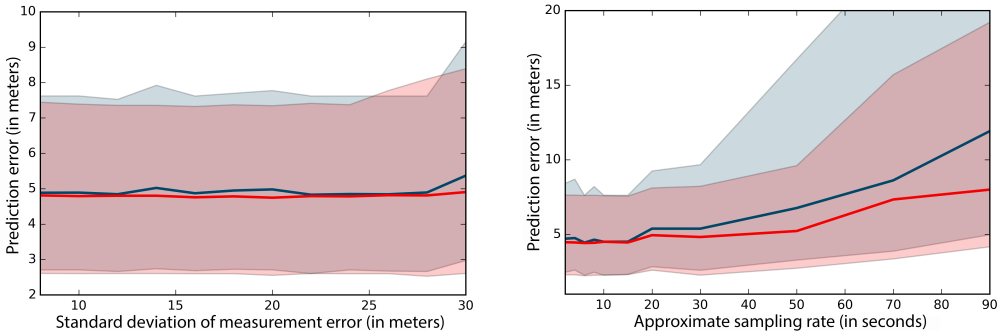
(a) most likely **(b)** second most likely

Figure 4.12: Example map-matching outcome with colour-coded probability scores for the two most probable paths.

The applicability of the algorithm to other datasets is then tested by artificially reducing the sampling rate of the data (removing some GPS points) and by increasing the noise of the data (perturbing GPS points). The algorithm shows good robustness against variation of the measurement noise (Figure 4.13a) that might in reality be due to high buildings, weather, etc.. However, it performs poorly on datasets with low sampling rates (Figure 4.13b). The decreased performance can be explained by the fact that low sampling rates largely increase the number of possible paths that the vehicle could have taken between subsequent GPS measurements (too many to cover with a fixed number of particles). The decrease in the algorithm's performance is particularly apparent when compared to the relatively good performance of the state-of-the-art deterministic approach, the ST-Matching algorithm.

In the next section, we introduce an improved particle filter method, called look-ahead particle filter, which is specifically designed to improve the accuracy of particle filters in highly informative observation schemes, e.g. due to low data sampling rate. It is a highly accurate, yet fully probabilistic, map-matching algorithm for low-sampling-rate GPS trajectories.

Sensitivity to the number of particles: In addition to evaluating the sensitivity to data quality above, we also investigate the internal performance of the algorithm by measuring its sensitivity to the number of particles and the length of the input GPS



(a) measurement error **(b)** sampling rate

Figure 4.13: Sensitivity of Particle Filter (blue) and ST-Matching (red) to GPS measurement error and sampling rate represented as 25th, 50th and 75th percentiles of map-matching errors.

trajectory.

The algorithm's accuracy converges exponentially as the number of particles increases. Therefore, at some threshold point (roughly equal to thirty particles for our dataset, see Figure 4.14), the algorithm is no longer sensitive to the number of particles and a further increase to the number of particles has no apparent influence on the algorithm's accuracy. This is an important advantage as the computational cost of the method scales linearly with the size of the particle population.

On the negative side, particle filter seems to degenerate exponentially as the length of the input trajectory increases (see Figure 4.15). Degeneration means that the standard deviation of the estimated probability distribution over possible paths taken by the vehicle becomes inaccurately low. Indeed, some degree of degeneracy is inevitable. Every resampling step reduces the number of unique latent values. For this reason, any SMC algorithm will fail for long enough input trajectory for any finite sample size, N, in spite of the asymptotic justification.

The problem of degeneracy is magnified by a limited accuracy of the proposed transition probability. The distribution captures our prior beliefs on the next position of the vehicle given its current location. If the distribution is too far off the true next position, few particles are likely to transition to the proximity of the next GPS measurement and hence to be resampled and propagated further. The fewer distinct particles survive, the lower the standard deviation of the probability distribution

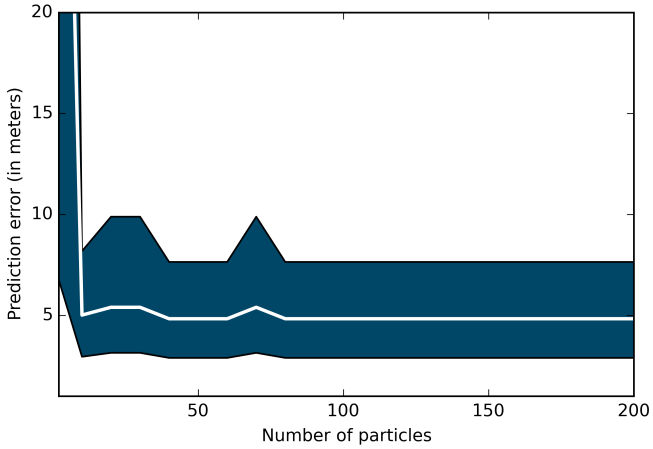


Figure 4.14: Sensitivity to the number of particles (25th, 50th and 75th percentile error).

over possible paths. The problem of degeneracy manifests itself in the decreased accuracy of the algorithm at lower sampling rates in Figure 4.13b.

The problem of inaccurate transition probability cannot be easily addressed within the standard particle filter framework as one of its requirements is that the transition probability depends on previous locations of the vehicle only. That is, it cannot ”look ahead” to make a more informed prediction of the vehicle’s location at the next timestamp. In the next section, we present our work on designing a new sequential sampling scheme that would outperform particle filter on datasets with low sampling rates, while still being fully probabilistic. The basic idea is to mimic some strengths of its deterministic counterpart, the ST-Matching approach, especially the ability to make informed guesses about possible true locations at each timestamp, but within a fully probabilistic framework. This should lead to increased accuracy and preserve the ability to calculate confidence estimates of map-matching outcomes.

4.5.3 Look-ahead particle filters

4.5.3.1 Overview

The ability to track a moving vehicle is of crucial importance in numerous applications. The task has often been approached by the importance sampling technique of particle filters due to its ability to model non-linear and non-Gaussian dynamics,

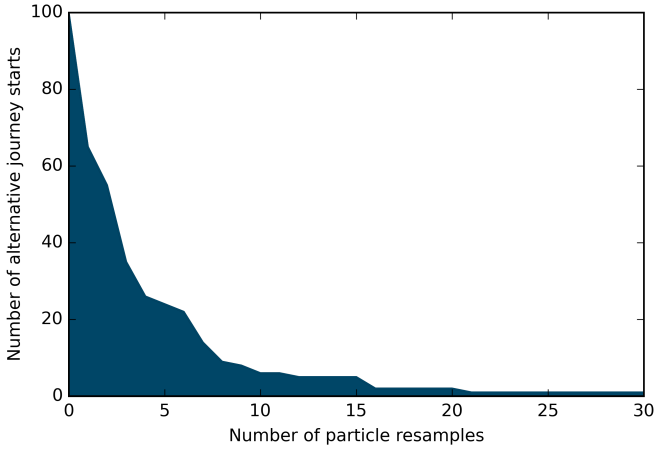


Figure 4.15: Degeneracy as the number of alternative journey starts represented by particles.

of which a vehicle travelling on a road network is a good example. As experienced in the previous Section 4.5.2, particle filters perform poorly when observations are highly informative. In this project, we address this problem by proposing particle filters that sample around the most recent observation. The proposal leads to an order of magnitude improvement in accuracy and efficiency over conventional particle filters, especially when observations are infrequent but low-noise.

4.5.3.2 Introduction

Tracking a moving vehicle is a central and difficult problem arising in different contexts ranging from military applications to robotics [60, 62]. It consists of computing the best estimate of the vehicle's trajectory based on noisy sensor measurements. In this research, we are interested in vehicle tracking when the road network is known.

Several strategies have been developed to track a vehicle on a road network [2, 63, 64, 65, 66, 67]. We focus on the particle filter method [28]. The method has had numerous successes in this area due to its flexibility to handle cases where the dynamic and observation models are non-linear and/or non-Gaussian. It is an importance sampling technique that approximates the target distribution by sampling from a series of intermediate proposal distributions.

Critically, in common with any important sampling method, the performance

of particle filters is strongly dependent on the choice of the proposal distribution. If the proposal is not well matched to the target distribution, then the method produces samples that have low effective sample size and, as a result, it requires a prohibitively large number of particles to represent the target distribution accurately. The problem typically arises under highly informative observation regimes, in which the current observation provides significant information about the current state but the state dynamics are weak.

The particle filter community has developed various approaches to mitigate the deficiency. One approach attaches a post-sampling step that moves particles sampled from the proposal distribution towards the target distribution using Markov Chain Monte Carlo moves [31, 32, 33] or by solving partial differential equations [34, 35, 36, 37]. An alternative approach improves the proposal distribution by giving it additional information about the current [38, 39, 40] or even future observations [41] or their approximations [42]. Several authors considered conditioning the proposal distribution on the current observation only [43, 32]. The approaches successfully increased the effective sample size, but, often at the cost of high computational complexity or analytical intractability. The construction of good, but also computationally efficient proposal distributions is still an open research question.

In this section, we propose an improved particle sampling scheme that is both computationally efficient and mathematically robust. The approach generates proposals based on the current sensor observation only, leading to good alignment between the proposal and the target distribution even with a small sample size. It converges to the desired target distribution at faster rates than standard particle filters, especially when observations are highly informative, e.g. infrequent but low-noise. It is easy to implement and avoids the computational and analytical complexity of the discussed alternatives with other proposal distributions or post-sampling moves. It also presents a simpler approach to sample weighing than those previously proposed with the same proposal distribution [43, 32].

The section is structured as follows. We present the problem statement in Section 4.5.3.3, focusing on the limitations of standard particle filters. We then

introduce the improved particle filters method in Section 4.5.3.4. We outline the application of the method to vehicle tracking in Section 4.5.3.5 and present results in Section 4.5.3.6. We conclude by summarising the project's contributions in Section 4.5.3.7.

4.5.3.3 Problem statement

The key idea of particle filters, introduced in Section 3.3.3, is to estimate the *target* posterior distribution $\pi(x_{1:N}) = p(x_{1:N} \mid y_{1:N})$, where $x_{1:N}$ is the state of the system until time N and $y_{1:N}$ is a sequence of measurements collected up to time step N.

In the context of vehicle tracking, the target posterior is our estimate of the trajectory, represented as the sequence of positions, followed by the vehicle until time N given all measurements collected until then. The measurements include *GPS readings* of latitude and longitude and *timestamps*.

Particle filters approximate the target distribution $\pi(x_{1:N})$ by a set of K weighted samples distributed according to $\pi(x_{1:N})$:

$$\pi(x_{1:N}) = \{x^{(i)}, w^{(i)}\}_{i=1,\dots,m} \tag{4.7}$$

where each $x^{(i)}$ is a sample (a state) and $w^{(i)}$ are non-negative weights called *importance factors* that determine the importance of each sample.

For what follows, it is important to remember that particle filter is an importance sampling scheme (see Section 3.3.3 for reference). That is, the samples $x^{(i)}$ approximating $\pi(x_{1:N})$ come from a (presumably simpler) proposal distribution $q(x_{1:N}|y_{1:N})$ and their weights $w^{(i)}$ are calculated according to the weight function

$$w(x_{1:N}) = \frac{p(x_{1:N}, y_{1:N})}{q(x_{1:N}|y_{1:N})}. \tag{4.8}$$

The proposal $q(x_{1:N}|y_{1:N})$ and the weight function $w(x_{1:N})$ are factorised according to (3.19) and (3.20) respectively, such that the estimation can be done recursively. At timestamp n, we estimate $\pi(x_{1:n})$ by sampling from an intermediate proposal distribution $q_n(x_n|x_{1:n-1}, y_{1:n})$ and by weighing the samples according to

an intermediate weight function

$$w_n(x_{1:n}) = \frac{p(x_n|x_{1:n-1})p(y_n|x_{1:n}, y_{1:n-1})}{q_n(x_n|x_{1:n-1}, y_{1:n})}. \tag{4.9}$$

In most cases, we follow the Markov assumptions (3.29) about the dynamical system to conveniently simplify the weight function in (4.9) to

$$w_n(x_{1:n}) = \frac{p(x_n|x_{n-1})p(y_n|x_n)}{q_n(x_n|x_{n-1})}. \tag{4.10}$$

In the case of bootstrap particle filter, which is the most common particle filter algorithm and our baseline, the proposal distribution is defined as the prior

$$q_n^{PF}(x_n|x_{1:n-1}, y_{1:n}) = p(x_n|x_{1:n-1}) \tag{4.11}$$

Subsequently, the importance weights in (4.10) become

$$\begin{aligned} w_n^{PF}(x_{1:n}) &= \frac{p(x_n|x_{n-1})p(y_n|x_n)}{q_n(x_n|x_{n-1})} \\ &= \frac{p(x_n|x_{n-1})p(y_n|x_n)}{p(x_n|x_{1:n-1})} \\ &= p(y_n|x_n). \end{aligned} \tag{4.12}$$

As shown in the preceding section, bootstrap particle filters is a popular approach to vehicle tracking and off-line map-matching. It shows high performance on noisy but frequent GPS trajectories. However, it radically decreases in accuracy on less frequent tracking data (see Figure 4.13b). In this section, we outline an improved particle filter algorithm that is designed to tackle this inefficiency by changing the proposal distribution $q_n(x_n|x_{1:n-1}, y_{1:n})$ used for sampling candidate positions at each timestamp.

4.5.3.4 Improved Particle Filters

We propose an improved particle filter (IPF) scheme in which x_n are sampled directly around the most recent observation y_n according to the proposal distribution:

$$q_n^{IPF}(x_n|x_{1:n-1}, y_{1:n}) = p(y_n \mid x_n) \tag{4.13}$$

This new proposal distribution possesses orthogonal strengths to the one in Equation 4.11, in that it generates samples that are highly consistent with the most recent sensor measurement but ignorant of past measurements. As such, we expect it to outperform conventional particle filters in systems where the current observation provides more information about the current state than the underlying state dynamics.

The incremental importance weights for these samples are again calculated by the quotient:

$$\begin{aligned} w_n^{IPF}(x_{1:n}) &= \frac{p(x_n|x_{n-1})p(y_n|x_n)}{q_n^{IPF}(x_n|x_{n-1})} \\ &= \frac{p(x_n|x_{n-1})p(y_n|x_n)}{p(y_n|x_n)} \\ &= p(x_n|x_{n-1}). \end{aligned} \tag{4.14}$$

Since x_{n-1} is represented by a set of samples $x_{n-1}^{(i)}$ weighted by importance factors $w_{n-1}^{(i)}$, the (non-normalised) importance factor for any sample $x_n^{(j)}$ can be approximated by

$$\sum_{i=1}^{K} p(x_n^{(j)} \mid x_{n-1}^{(i)}) w_{n-1}^{(i)} \tag{4.15}$$

The importance weights reflects the likelihood of the sample given *past* measurements. This is orthogonal to the bootstrap particle filter in Equation 4.12, where it depends on the *current* measurement only. This is also different from the bootstrap particle filter in that it uses the entire population of particles at time $n-1$ instead of a single ancestor $x_{n-1}^{(i)}$ to propose a weighted particle $x_n^{(j)}$ at time n. Since now we propose samples without "looking back", we assume that the new particles are equally likely to be descendants of any previous particles.

Overall, the proposed sampling scheme changes how data are used in belief estimation: the current measurement is now used for sampling (instead of weigh-

ing); past measurements are used for calculating importance factors (instead of sampling). The sampling scheme mimics some strengths of the probabilistic ST-Matching algorithm introduced in Section 4.5.1, especially the ability to make informed guesses about possible true locations at each timestamp. It extends this idea from sampling the possible location from a fixed candidate set (Figure 4.5a) to a continuous probability distribution over the road network (Figure 4.16). The extension should improve map-matching accuracy by increasing the range of possible map-matching solutions.

The scheme is implemented recursively according to Algorithm 2.

Algorithm 2 Look-ahead particle filters for state space Markov models.

- **Initialisation**: At time $n = 1$, draw K particles according to initialisation probability $p(y_1|x_1)$. Call this set of particles X_1.
- **Recursion**: At time $n > 1$, generate K particles by sampling from the observation probability $p(y_n \mid x_n)$. Call the resulting set $\overline{X}_n$. Subsequently, draw K particles (with replacement) with a probability proportional to the importance weight in (4.15). The resulting set of particles is X_n.

In the context of vehicle tracking, the particles stored in X_n approximate the vehicle *position* at time n. If instead of the single-time approximation, you are interested in finding the most likely *trajectory* that the vehicle traversed until time n, it can be computed via the following dynamic programming routine. It corresponds to finding the sequence $x_{1:n}$ that maximises the posterior $p(x_{1:n} \mid y_{1:n})$.

1. Choose a sample $x_n^{(i)}$ from the sample set $\overline{X}_n$ that has the highest importance factor $w_n^{(i)}$.
2. Use the sample $x_n^{(i)}$ to find a preceding sample $x_{n-1}^{(j)}$ from $\overline{X}_{n-1}$ that maximises $p(x_n^{(i)} \mid x_{n-1}^{(j)}) w_{n-1}^{(j)}$, i.e. is the most likely preceding state. Repeat this step until you reach $n = 1$.

4.5.3.5 Application to Vehicle Tracking

Implementation: In order to apply the improved particle filters to vehicle tracking, one must specify the form of the observation probability $p(y_n \mid x_n)$ and the transition

probability $p(x_n \mid x_{n-1})$. Their forms depend on the vehicle's dynamics and the type of sensor used for localisation (a GPS receiver in this case). The distributions are time-invariant; hence we will omit the time index n in the following derivations.

A **Observation probability**

Definition: We model the conditional probability $p(y \mid x)$ of observing a GPS point $\boldsymbol{y}$, represented by its easting and northing coordinates:

$$\boldsymbol{y} = \begin{pmatrix} y_e \\ y_n \end{pmatrix} \tag{4.16}$$

as a two-dimensional Gaussian distribution

$$\boldsymbol{y} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$$

with the mean vector $\boldsymbol{\mu}$ representing the true vehicle position $\boldsymbol{x}$

$$\boldsymbol{\mu} = \boldsymbol{x} = \begin{pmatrix} x_e \\ x_n \end{pmatrix} \tag{4.17}$$

and the covariance $\boldsymbol{\Sigma}$ that is constant across space, i.e. *isotropic* covariance

$$\boldsymbol{\Sigma} = \begin{bmatrix} \Sigma_{ee} & \Sigma_{en} \\ \Sigma_{ne} & \Sigma_{nn} \end{bmatrix} = \begin{bmatrix} \sigma^2 & 0 \\ 0 & \sigma^2 \end{bmatrix} \tag{4.18}$$

This representation of $p(y \mid x)$ reflects our expectation that GPS observations are normally distributed around the true vehicle positions.

Proposal Generation: In the proposed method, we use the observation probability $p(y \mid x)$ to sample possible vehicle positions on the road network (see Equation 4.13). Since $p(y \mid x)$ is defined as a two-dimensional Gaussian, this corresponds to sampling positions on the road network from a two-dimensional Gaussian centred at each observation, as shown in Figure 4.16.

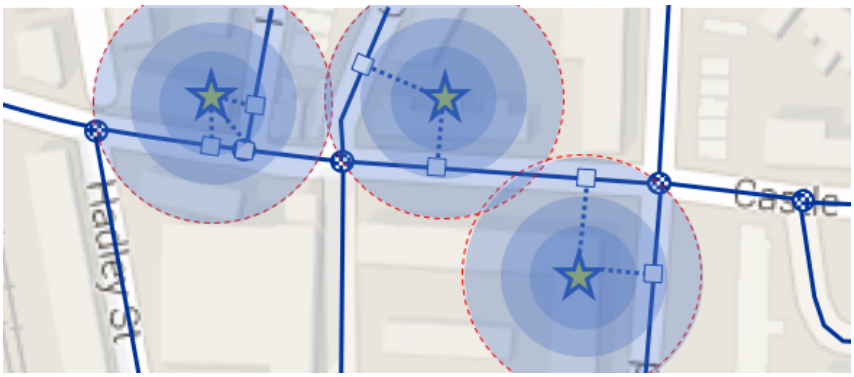


Figure 4.16: Candidate positions for each GPS observation are sampled from a Gaussian distribution (shaded in blue) around the observation.

In the previous work in Section 4.5.2, we followed a naive sampling approach in which we sampled positions from a two-dimensional Gaussian and then removed the samples if they did not coincide with the road network. Here, we develop a more efficient approach for generating samples on the road network only (as shown in Figure 4.11b) by analytically projecting the two-dimensional $p(y \mid x)$ onto individual road segments.

We begin with the general form of a two-dimensional Gaussian distribution for $p(y \mid x)$

$$p(y \mid x) = \mathcal{N}(\boldsymbol{y} \mid \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{2\pi|\boldsymbol{\Sigma}|^{1/2}} exp\left\{-\frac{1}{2}(\boldsymbol{y}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{y}-\boldsymbol{\mu})\right\} \quad (4.19)$$

We precompute the inverse of the covariance matrix

$$\boldsymbol{\Sigma}^{-1} = \frac{1}{\sigma^4}\begin{bmatrix}\sigma^2 & 0\\ 0 & \sigma^2\end{bmatrix} = \begin{bmatrix}\sigma^{-2} & 0\\ 0 & \sigma^{-2}\end{bmatrix}$$

and use it together with the partitioning (4.16), (4.17), and (4.18) to rewrite (4.19) as

$$
\begin{aligned}
p(y \mid x) &= \frac{1}{2\pi\sigma^2} exp\left\{-\frac{1}{2}\left[\frac{(y_e-\mu_e)^2}{\sigma^2}+\frac{(y_n-\mu_n)^2}{\sigma^2}\right]\right\} \\
&= \frac{1}{(2\pi\sigma^2)^{1/2}} exp\left\{-\frac{1}{2\sigma^2}(y_e-\mu_e)^2\right\} \times \frac{1}{(2\pi\sigma^2)^{1/2}} exp\left\{-\frac{1}{2\sigma^2}(y_n-\mu_n)^2\right\} \\
&= \mathcal{N}(y_e \mid \mu_e, \sigma) \times \mathcal{N}(y_n \mid \mu_n, \sigma)
\end{aligned}
\tag{4.20}
$$

We successfully factor $p(y \mid x)$ into a product of two Gaussian distributions along the *easting* and *northing* directions due to the isotropic properties of the covariance matrix in (4.18). In fact, the factorisation of $p(y \mid x)$ holds for any other orthogonal coordinate system. Therefore, we replace the easting-nothing coordinates with orthogonal distances from x dictated by the road segment that x is on: a (distance *to* the road segment), b (distance *along* the road segment).

Under the new coordinate system x and y are partitioned as

$$
x = \begin{pmatrix} x_a \\ x_b \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \qquad y = \begin{pmatrix} y_a \\ y_b \end{pmatrix}
$$

and $p(y \mid x)$ becomes

$$
\begin{aligned}
p(y \mid x) &= \mathcal{N}(y_a \mid \mu_a, \sigma) \times \mathcal{N}(y_b \mid \mu_b, \sigma) \\
&= \mathcal{N}(y_a \mid 0, \sigma) \times \mathcal{N}(y_b \mid 0, \sigma)
\end{aligned}
\tag{4.21}
$$

Since the new coordinate system is centered on the true position x, the mean distances μ_a and μ_b above are set to zero. The observation probability becomes a product of probabilities of the distance y_a orthogonal to the road segment and the distance y_b along the road segment where x is.

The above definition enables us to generate proposals x in accordance with

the observation model $p(y \mid x)$ (as specified in Equation 4.13):

(a) Firstly, sampling a road segment that x in on such that $y_a \sim \mathcal{N}(0, \sigma)$.

(b) Secondly, sampling the position of x along the segment such that $y_b \sim \mathcal{N}(0, \sigma)$.

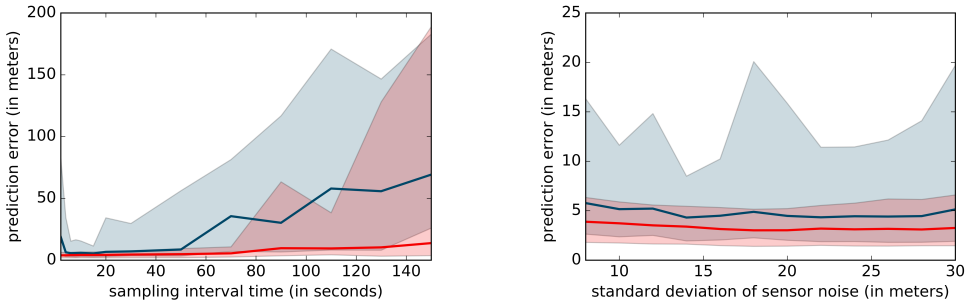
B **Transition probability**

We set the transition probability $p(x_n \mid x_{n-1})$ to be a linear estimate equal to the Cartesian distance between GPS points x_{n-1} and x_n plus an additive Gaussian noise. This is the same simplistic assumption that we used in the bootstrap particle filter in Section 4.5.2 that could be further explored, but it is not the focus of this project.

4.5.3.6 Results

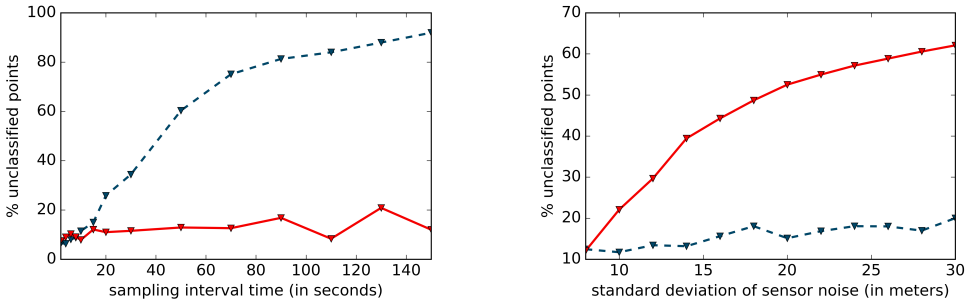
A series of tests was conducted to elucidate the difference between the standard and the proposed particle filters on a range of GPS trajectories constructed according to the validation framework in Section 4.4. We found that the modified proposal distribution consistently outperforms conventional particle filters in terms of accuracy. As expected, largest gains in accuracy are observed on datasets with long sampling intervals as their observations are infrequent and hence become highly informative. Figure 4.17 plots the prediction error (in meters) of both algorithms for different sampling intervals and levels of sensor noise, using $K = 10$ samples only. It shows that the proposed method has lower *median* error across all examined sampling rates and sensor noise levels, as well as much lower error *variation*.

We evaluated the ability of both methods to track a vehicle over time. When they fail to track a vehicle, it means that all positions that they propose are completely unlikely given sensor data, i.e. unnormalised particle weights sum up to zero. The standard particle filter basically fails when sensor measurements are infrequent (with $K = 10$ samples). Figure 4.18a shows that it is unable to track the vehicle nearly 70% of the time when the sampling interval increases to one minute. In the same scenario, the proposed method gives excellent results that show little variation to changes to sampling intervals.



(a) sampling rate **(b)** sensor noise

Figure 4.17: Accuracy of the improved particle filters (red) and the standard particle filters (blue) on GPS data with varied sampling rate and sensor noise, represented as 25th, 50th and 75th percentiles of prediction errors.

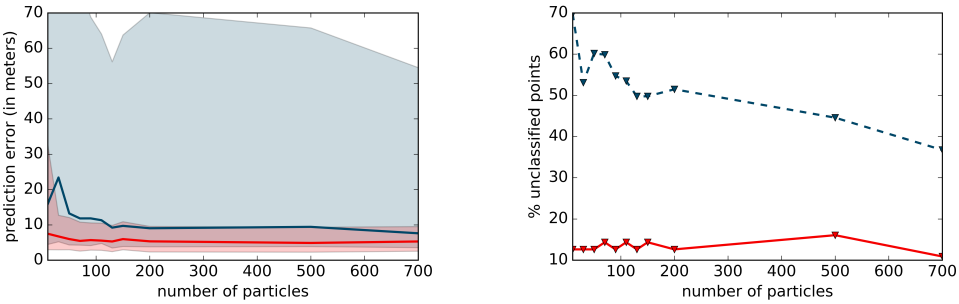


(a) sampling rate **(b)** sensor noise

Figure 4.18: Percentage of time the improved particle filters (red) and the conventional particle filters (blue) lost track of the position of the vehicle as a function of the GPS sampling rate and the sensor noise.

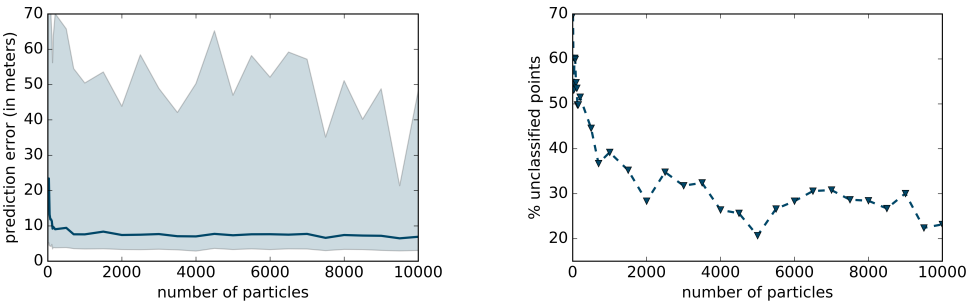
On the contrary, the proposed method fails to track when sensors are very noisy. Although it shows high accuracy (see Figure 4.17b), it is prone to high failure rates as the level of sensor noise increases (Figure 4.18b). This weakness reflects the orthogonal limitations of the two approaches: our method generates samples that are highly consistent with the most recent measurement (which makes it sensitive to sensor noise), whereas the conventional approach samples in accordance with past measurements (inefficient when sampling rates are low).

Finally, we tested the sensitivity of the proposed method to the number of samples used. Figure 4.19 shows comparative results on GPS data with the sampling interval of 70 seconds. The proposed method yields significantly better results, both in terms of accuracy and robustness to failure. When only $K = 10$ samples are used,



(a) accuracy **(b)** failure rate

Figure 4.19: Accuracy and robustness of the improved particle filters (red) and the conventional particle filters (blue) as a function of the number of samples used. Accuracy is shown as the 25th, 50th and 75th percentiles of prediction errors.



(a) accuracy **(b)** failure rate

Figure 4.20: Accuracy and robustness of the standard particle filters as the number of samples is increased to very large values. Accuracy is shown as the 25th, 50th and 75th percentiles of prediction errors.

it reduces the estimation error by almost 10 meters and the percentage of failure by as much as 68%. The performance is further improved when more samples are used, but the gain is small compared to the conventional particle filters. In fact, the proposed method with $K = 100$ samples is more accurate and robust than the conventional particle filters with as many as $K = 10000$ samples (see Figure 4.20). Therefore, it can be reliably used with a small number of samples, making it highly computationally efficient.

4.5.3.7 Discussion

This piece of research designs a modified particle filter method that shows uniformly superior accuracy to the conventional particle filters. The improved algorithm utilizes a different proposal distribution which uses only the most recent observation in the position prediction process. In doing so, it makes more efficient use of the particles, particularly in situations in which the transition noise is high in relation to the observation noise.

The main contribution of this research is the proposal distribution itself and the derivation of the associated importance weights that guarantees convergence to the same posterior distribution as the standard particle filters. An important contribution is also the projection of a two-dimensional Gaussian onto a network of roads, which enables efficient sampling on the road network from a spatial Gaussian.

The theoretical contributions are complemented by experimental results of vehicle tracking using a police GPS dataset. The new algorithm is consistently more accurate than the standard particle filters, with largest gains in accuracy on sparse GPS data. It requires much fewer samples to yield good performance. In fact, as few as fifty samples are sufficient to outperform the standard method with 10,000 particles in terms of accuracy and proneness to failure. We believe that our results illustrate that particle filters can be radically improved if one carefully chooses a proposal distribution, so it extracts the most information from the available data.

4.6 Conclusions

The chapter proposes novel approaches to improved localisation in cities using the process of map-matching. The approaches are examples of model-based machine learning which use a state space hidden Markov model to represent the relationship between true and observed locations. Subsequently, they learn the true locations from noisy tracking data by making inference in the Markov model.

Despite using the same graphical model, the proposed methods differ in their computational efficiency and flexibility. Broadly speaking, methods that offer little flexibility, such as PST-matching that can only snap data points to the closest points

on roads within a fixed search radius, tend to be computationally efficient and vice versa.

The chapter makes and initial attempt at designing a map-matching method that is *both* flexible and efficient. In particular, look-ahead particle filter inherits the fine-grained accuracy of particle filter while showing orders of magnitude improvements in efficiency when tracking data are sparse. The computational gains are only possible by exploiting the structure of data in method design. Future work could focus on methods which can self-tune for high efficiency and accuracy. An initial attempt at such an automated approach is our work on adversarial particle filter in Chapter 6, where the method learns a highly flexible proposal model directly from historical data.

Chapter 5

How you navigate: extracting urban regions from routing data

5.1 Overview

Do administrative boundaries correspond to the observable ways in which people interact in urban space? As cities grow in complexity, and people interact over long distances with greater ease, so partitioning of cities needs to depart from conventional gravity models. The current state-of-the-art for uncovering *interactional* regions, i.e. regions reflective of observable human mobility and interaction patterns, is to apply community detection to networks constructed from vast amounts of human interactions, such as phone calls or flights. This approach is well suited for origin-destination activities, but not for activities involving multiple locations, such as police patrols, and is blind to spatial anomalies. As a result of the latter, community detection generates geographically coherent regions, which may appear plausible but give no insights into forces other than gravity that shape our interaction patterns.

This chapter proposes novel approaches to regional delineation that address the aforementioned shortcomings. Firstly, it introduces topic modelling, an example of model-based machine learning, as an alternative tool for extracting interactional regions from tracking data. Secondly, it presents refinements of the topic modelling and community detection approaches that can uncover interaction patterns driven by

forces other than spatial proximity. When applied to police patrol data, the proposed methodology partitions the street network into non-overlapping patrol zones and detects popular long-distance routes between police stations. Finally, the chapter applies the discovered regions to the problem of route prediction. Initial results show improved prediction accuracy in comparison to an alternative route prediction approach based on the shortest-distance assumption. These findings could be used in the design of effective police districts, especially in light of recent funding cuts that promise to impact upon the ways in which policing and specifically patrols are carried out.

5.2 Introduction

Cities are ”not simply places in space but systems of networks and flows” [1]. As such, they represent highly structured and dynamic environments that provide the loci of human mobility and interaction. The structure of cities both shapes and is shaped by patterns of human interactions, and hence urban analytics should be founded upon areal units that reflect such patterning.

To this end we propose the concept of *interactional regions* which reflect the ways in which people are observed to move and interact. Interactional regions are spatial envelopes that commonly bound human activities and interactions, such as consumer transactions, taxi routes or police patrols. They respect the natural ways in which people interact across space and, as such, their definition is essential for effective business and service planning, including the assignment of administrative responsibilities in public resource allocation.

Administrative geographies are inevitably an uneasy compromise between existing and past patterns of spatial interaction, with the latter encapsulated in so-called ’place effects’ [68]. From this perspective, places can themselves be construed as the accretion of past interactions, making places unique, but nonetheless comparable with others that have interactional histories that may be similar in different ways. Boundaries may have been created many decades ago, when human interactions and mobility were predominantly local and the conceptual separation

of human populations into fixed and geographically coherent regions was plausible and useful. However, the accelerating scale and pace of societal evolution combined with observable changes in the frictions of distance result in new, multifaceted and increasingly complex patterns of human connectivity [69]. It is these that nevertheless define contemporary interactions between established places [70]. Spatial interaction patterns are no longer a simple manifestation of the distance attenuation functions of traditional gravity models (if they ever were) or of opportunity functions of radiation models [71], but of a far more complex range of interacting factors that can only be uncovered by analysing vast amounts of human-generated flow data, such as phone calls [72, 73], monetary transactions [74, 75] or vehicle flows [76, 77].

In practice, the current state-of-the-art for identifying interactional regions is to apply community detection techniques to the flow networks created by aggregating human flows between locations [72, 77, 73]. This approach enables analysis of interactions without the geographical presupposition inherent to gravitational models. However, it has two important limitations. Firstly, it is designed for datasets with clearly defined origins and destinations for each interaction. Examples of such datasets include phone calls (with caveats), taxi journeys or retail transaction data. Counterexamples include continuously generated data, such as tracking data from police vehicles or mail delivery vans, where journey origins and destinations are not functionally defined or known.

Secondly, and most importantly, community detection is shaped by the pre-existing spatial structure of settlements [78, 79]. In most cases, it uncovers regions that are strongly determined by geographical proximity at the expense of other underlying forces shaping the interactions. For instance, traffic flows are typically dominated by low-cost short-ranged interactions. As a result, community detection is blind to spatial anomalies and only identifies regions which are compact in physical space. This leads us to the central question of our work: can we detect interaction patterns that build upon more than distance attenuation? In other words, if we control for gravity-like forces, what other forces shape our interaction and mo-

bility patterns? And can we develop a standard network methodology to uncover them?

In this chapter, we propose novel approaches to regional delineation that address the above limitations. Firstly, we propose a method of topic modelling for extracting interactional regions from new forms of data, i.e. tracking data with no origin and destination specified. Secondly, we extend community detection and topic modelling to uncover interaction regions driven by forces other than spatial proximity. We factor out the effect of space in order to reveal more clearly hidden interaction patterns between places. Finally, we apply the discovered regions within a region-based route prediction framework.

We validate our methodology using GPS traces from police patrol vehicles in the London Borough of Camden. Our data are derived from a wider investigation into the local geography of criminal activity and proactive initiatives by police and citizens to reduce crime. The data are particularly relevant to introducing the concepts described in the chapter because of the requirement to patrol the all street segments in the study area.

5.3 Proposed models

Our methodology presents a comprehensive suite of algorithms for extracting interactional regions from large volumes of mobility tracking data in networked environments, such as cities. The data are given as sequences of observations, each corresponding to an episode of mobility or a *journey*. The methodology is motivated by data generated by police patrol vehicles, but is equally applicable to any tracking data that pertain to separable episodes.

At the base of the methodology is a representation of tracking data as a flow network. Interactional regions are extracted as patterns on that network using two clustering approaches: community detection and topic modelling. Community detection assigns locations to regions given flows between *pairs of locations*, hence treating tracking data in an origin-destination fashion, whereas topic modelling mines flow patterns from *location sequences* corresponding to complete journeys.

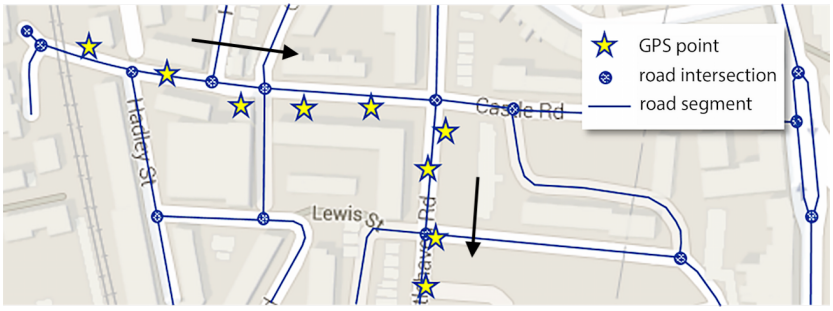


Figure 5.1: Illustration of a GPS trajectory generated by a vehicle travelling in the direction indicated by the black arrows.

The two methods lead to different definitions of interactional regions, which will be clearly stated in the following subsection.

Finally, we propose novel extensions of community detection and topic modelling that enable us to uncover spatially anomalous interaction patterns. By accounting for spatial forms of cities, we answer the central question motivating this work: can we detect interaction patterns that are not due to space? The obtained *spatially-independent* interactional regions augment the more traditional view on interactional regions obtained from standard community detection and topic modelling techniques [72, 77, 73] which do not disentangle spatial effects from other effects of interest.

5.3.1 Flow network

We begin the flow network creation by mapping vehicle traces, in the form of sequences of GPS observations, to the underlying street network. We perform map-matching using the technique of ST-Matching[1] proposed by [2]. The technique converts complete GPS traces of vehicle journeys into sequences of visited street segments (see an example GPS trace in Figure 5.1 and its map-matching output in Table 5.1). Each street segment is a piece of road, not necessarily straight, between two neighbouring road intersections and is represented by a unique identifier provided by [80].

We construct the flow network by representing street intersections as network

[1]Further work could replace ST-Matching with our improved methods in Chapter 4, however, the methods had not yet been finalised when this research was underway.

No.	GPS ping (easting, northing)	Street segment (id)
1	(529830.826879,182824.079602)	osgb4000000030239207
2	(529901.218657,182821.928288)	osgb4000000031283337
3	(529982.352018,182809.024134)	osgb4000000030250373
4	(530037.990277,182811.410456)	osgb4000000031283336
5	(530101.259824,182812.320039)	osgb4000000031283336
6	(530183.132569,182801.38203)	osgb4000000031283326
7	(530176.143972,182778.158761)	osgb4000000031283326
8	(530181.99883,182739.597688)	osgb4000000031186773
9	(530178.955762,182707.345447)	osgb4000000031186773

Table 5.1: Example conversion from a GPS sequence (mapped in Figure 5.1) to a sequence of visited street segments. The order of GPS pings reflects the direction of travel in Figure 5.1.

nodes (vertices) and vehicle visits to street segments as undirected network edges. This definition allows multiple edges between a pair of nodes, each corresponding to a single visit to the underlying street segment. We remove nodes that have no edges as they do not provide any information on mobility patterns.

5.3.2 Interactional regions as communities

Our first approach to interactional region extraction is community detection. In network science, community detection refers to the problem of finding the natural divisions of a network into groups of vertices, called *communities*, such that there are many edges within groups and few edges between groups [81].

In our context, community detection mines interactional regions from the flow network as groups of highly interconnected street intersections (nodes). Thus, it leads to the following network-based definition of interactional regions:

Definition 5.3.1 ***(Interactional regions as communities)*** *An interactional region is a collection of street segments that have high volumes of traffic flow between them.*

Community detection in its basic form is often used by the GIS community for uncovering interactional regions from flow networks. We begin by introducing the basic community detection algorithm and then propose novel modifications that make the method better suited for spatial analysis.

5.3.2.1 Standard approach or current state-of-the-art

What is meant by ”few edges between groups” and ”many edges within groups” in community detection is debatable and different definitions have led to a variety of algorithms for community detection. The most common formulation of the problem, and the one adopted in this chapter, is of modularity optimisation.

Modularity is a measure of the quality of a network partition, which has a high value when more edges in a network fall *within* rather than *between* communities. In practice, the current state-of-the-art for finding modules in spatial networks is to optimize the standard Newman-Girvan modularity [82, 83], which assigns vertices to the same community if there are more edges between them than one would expect were edges simply placed at random in the network. In the next section, we will argue that this approach overlooks the spatial nature of the system, or the city as it is in this case.

Modularity is formally defined as:

$$Q = (\text{fraction of edges within communities}) - (\text{expected fraction of such edges}) \tag{5.1}$$

It considers fractions of edges rather than absolute counts hence it is unaffected by the total number of edges in the network. In mathematical terms, modularity score reads:

$$Q = \frac{1}{2m} \sum_{C \in \mathscr{P}} \sum_{i,j \in C} [A_{ij} - P_{ij}] \tag{5.2}$$

where $i, j \in C$ is a summation over pairs of nodes i and j belonging to the same community C of a network partition into communities $\mathscr{P}$ and therefore counts edges within communities. A is the adjacency matrix storing the observed number of edges A_{ij} between nodes i and j and P is a matrix storing the expected number of edges between any two nodes. Our estimate of the expected number of edges depends on our null model. The most popular null model, proposed by [84], is:

$$P_{ij} = k_i k_j / 2m \tag{5.3}$$

where $k_i = \sum_j A_{ij}$ is the degree of node i. Finally, modularity Q is normalized by the total number of edges in the network $m = \sum_{i,j} A_{ij}/2$.

Equation 5.3 defines that, under the Newman-Girvan (NG) null model, the expected number of edges between any two nodes is proportional to the product of the degrees of the nodes. That is, the more edges a node has, the more likely it is to connect to a different node in the network. Although this definition makes intuitive sense, it overlooks any underlying constraints that might impact on edge formation in spatially-embedded networks, such as spatial distance. We will address this limitation in the following section.

Modularity optimization is a computationally hard problem [81]. Algorithms that guarantee to find network partitioning with maximum modularity take exponentially long to run and hence are only useful for synthetically small networks [85]. Instead, therefore, we turn to a *heuristic* algorithm, an algorithm that *approximates* the optimal modularity in an efficient way. We use a popular heuristic algorithm known as the Louvain Method of community detection [86].

Louvain Method scales well to large networks and is capable of clustering networks with weighted edges. It is advantageous in that users can easily modify the definition of modularity that it aims to maximise. This characteristic will be particularly useful when we introduce a spatial adaptation of the NG modularity in the next section.

Louvain Method approaches an optimal partition of a network into communities by first assigning each node to a different community and then iteratively merging communities into partitions that increase the overall modularity score Q. The algorithm converges when no further aggregation is found to increase the score.

5.3.2.2 Spatial communities

The standard approach to community detection presented in Section 5.3.2.1 assumes that there are no underlying constraints that could impact on the formation

of edges in our network. In other words, any clustering patterns that we observe are of interest to us. Unfortunately, this does not often hold true in practice, where we might want to exclude obvious patterns from those of interest in our analysis.

In our case, these obvious patterns are due to spatial proximity or, more precisely, the configuration of the underlying street network. We are bound to observe movement in our flow network between nodes that in reality are endpoints of the same street segment. On the other hand, we can be quite certain that there will be no direct traffic between nodes that have no connecting road segment. In this section, we propose a way of disentangling these effects from our clustering results in order to discover interaction relations between places that arise not merely because of spatial adjacency.

We propose the following modification of the NG null model presented in (5.3):

$$P_{ij} \propto k_i k_j \cdot f(i,j) \quad \text{where} \quad f(i,j) = \begin{cases} 1, & \text{if nodes } i \text{ and } j \text{ are endpoints of the same street.} \\ 0, & \text{otherwise.} \end{cases} \tag{5.4}$$

The 0/1 function $f(i,j)$ incorporates our knowledge of the underlying street network. If two nodes are endpoints of the same street segment, then we retain the standard expectation proposed in (5.3) to reflect the fact that the more traffic passes through each node, the higher the chance that some of the traffic will occur between them. By contrast, if they are not directly connected, we reduce the expected flow between them to zero. Notice the proportionality sign in (5.4): once entries of P are calculated, P has to be renormalised to ensure that the total weight is conserved, i.e. $\sum_{ij} A_{ij} = \sum_{ij} P_{ij} = 2m$.

Intuitively, our proposal works by incorporating our knowledge of the street network into the calculation of the *expected* number of edges between nodes in the flow network. If nodes are endpoints of the same street segment, we expect some traffic between them and our expectation is uniform across all pairs of such nodes. The more traffic we observe, the more likely we are to put the nodes into the same

community, i.e. a group of nodes with *higher than expected* traffic between them.

This approach to spatial community detection is inspired by [79], who use a linear function akin to $f(i,j)$ to capture distances in Cartesian space between nodes on a flow network. Here we adapt this approach from Cartesian space to urban space, where interactions are influenced by the connectivity of the underlying street network. Further work could extend the proposed function $f(i,j)$ to account for subtler spatial effects, such as differences in expected flows on major and minor roads.

Our spatial modification impacts on the expected number of edges calculated in (5.2), thus changing the value of modularity Q for any given network partitioning. Despite the change to Q, the same iterative approach as in Section 5.3.2.1, the Louvain Method, can be used to find a partitioning of the flow network into interactional regions that maximises the modified Q. We implement our spatial community detection with a generalized Louvain Method proposed by [87].

5.3.3 Interactional regions as topics

Our second approach to the extraction of interactional regions is topic modelling. Similar to community detection, this is an approach to finding clusters in data. However, instead of extracting them from pairwise similarities between items, it detects clusters as repetitive themes in unstructured collections of items. The themes are represented as latent variables in a probabilistic graphical model and then inferred from data using an approximate inference sampler.

Topic modelling was originally developed to discover main themes that pervade a large collection of documents [88]. Loosely speaking, it defines a topic as a collection of words concerning a common subject. It assumes that documents can exhibit multiple topics and mines these topics from large collections of documents by detecting groups of words that repeatedly occur together (*co-occur*) in documents.

Since its conception, topic modelling has been adopted to handle many kinds of data, including audio and music, computer code and social networks. Here, we adopt topic modelling to deal with vehicle journey data in order to uncover

interactional regions. We assume that vehicle journeys can traverse one or more interactional regions (*topics*). We mine interactional regions from a large collection of vehicle journeys (*documents*), as groups of street segments (*words*) that often occur together in vehicle journeys. This underpins a second, alternative, definition of interactional regions proposed in this chapter:

Definition 5.3.2 ***(Interactional regions as topics)*** *An interactional region is a collection of street segments that often co-occur in journeys.*

Notice that the above definition of interactional regions differs slightly from Definition 5.3.1 in Section 5.3.2. It uses an extra level of information on interactions through the inclusion of complete vehicle journeys. This ensures that even spatially distant street segments in the same interactional region are related, since they often co-occur in vehicle journeys. This does not always hold true for community detection results, where interactional regions are formed based on pairwise relations between nodes. As a result, a *community* region might contain a pair of street segments because of high interactions between their neighbours, neighbours of their neighbours, etc., without any guarantee that a car has ever driven from one node to the other.

5.3.3.1 Latent Dirichlet Allocation

The most widely used topic model, and the one used here, is Latent Dirichlet Allocation (LDA) proposed by [89]. LDA is a *probabilistic model* that belongs to a family of generative probabilistic models. As introduced in Section 3.2.1, in generative probabilistic modelling, we treat our data as arising from a generative process that includes unobserved (hidden) variables. This generative process defines a joint probability distribution over the observed and the hidden random variables. We perform data analysis by using that joint distribution to compute the conditional distribution of the hidden variables given the observed variables. This conditional distribution is also called the posterior distribution.

LDA attempts to capture the notion that documents exhibit multiple topics. It defines a generative process from which documents could have arisen. It states

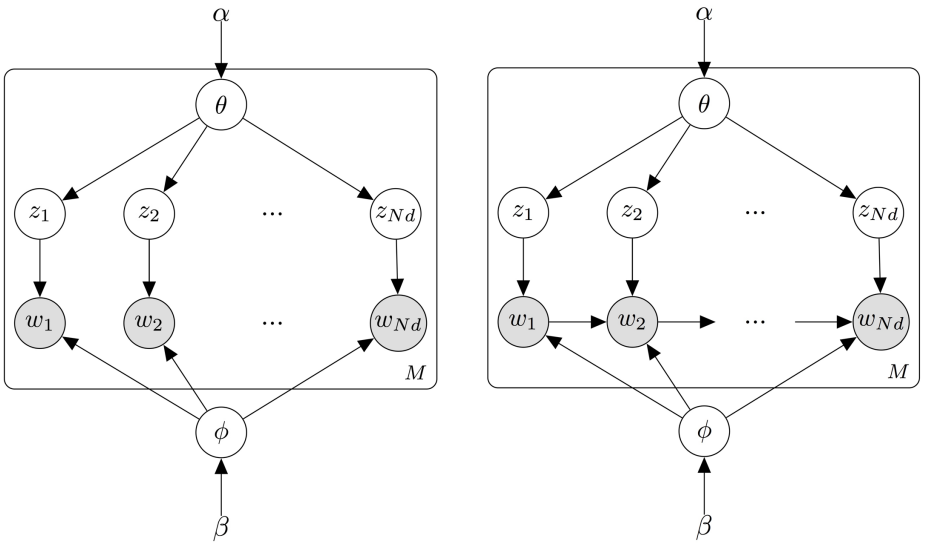
that the observed variables are the words of the documents; the hidden variables are the topics; and the generative process is as described here. The computational problem of inferring the hidden topic structure from the documents is addressed by the collapsed Gibbs sampler developed by [90]. It is a sampling-based algorithm that approximates the posterior distribution by a finite number of samples from it.

LDA is described more formally with the following notation. There are K pre-defined topics, each taking a probabilistic distribution over a fixed vocabulary. When documents $D = \{d_1, d_2, ..., d_M\}$ are generated, a topic mixture θ for each document is sampled, with $\theta_{d,k}$ indicating the topic proportion of topic k in document d, from a Dirichlet distribution with prior α. Subsequently, topics $Z_D = \{z_1, z_2, ..., z_{Nd}\}$ for each word in the document are sampled from that mixture. Finally, based on the sampled topics, words $W_D = \{w_1, w_2, ..., w_{Nd}\}$ are chosen from the topics' distributions ϕ over the vocabulary, where ϕ_k is the distribution of topic k over the vocabulary, sampled from a Dirichlet distribution with prior β. The graphical model for LDA is illustrated in Figure 5.2a.

Our core contribution is to adapt LDA to regional delineation problems by interpreting topics as interactional regions. The topics are inferred from large collections of vehicle traces (*documents*), where each trace is a sequence of visited street segments (*words*). The topics are represented as probabilistic distributions over all possible street segments. If we are interested in a binary partition of the street segments into topics, we assign each segment to the topic under which it has the highest probability.

5.3.3.2 Spatial topics

Conventional topic modelling is not well designed for spatial data. This is because the basic technique is not attuned to the nature of spatial data and hence cannot disentangle spatial patterns from other patterns that might be of more interest to the user. What is more, one of the assumptions of the core technique is that consecutive words within a document are independently sampled under the 'bag-of-words' assumption. This assumption is already simplistic for text but combined with the spatial ignorance, these characteristics limit the usefulness of topic modelling for



(a) LDA **(b)** Token-Bigram LDA

Figure 5.2: Graphical illustration of topic models for interactional region extraction, assuming (a) independence and (b) dependence between consecutive street segments w_i and w_{i+1} in observed journeys. Each node is a random variable and each edge indicates statistical dependence between the variables. The rounded rectangles denote replication for all M vehicle traces.

geographic problems in general and interactional region extraction in particular. Our documents, vehicle traces, are inherently spatial and sequential. We thus need to incorporate these qualities in the model in order to identify any significant patterns in the data.

Our contribution is to accommodate the properties of spatial data by introducing the notion of dependence between consecutive words in the generative process captured by LDA (see Figure 5.2b). Such dependence has previously been proposed by [91] as a token-bigram topic model, in which the dependence is interpreted as a transition probability $p(w_{i+1}|w_i)$, i.e. given an occurrence of word w_i, how likely is it that word w_{i+1} will occur next in the document sequence? In the original paper, the dependence was successfully deployed in text modelling. In our case study, the transition probability is derived directly from the branching of the underlying street network. That is, given that a vehicle is on street segment w_i, how likely it is to move to street w_{i+1}? The probability is zero for non-adjacent street segments and inversely proportional to the number of street segments adjacent to w_i, k_{w_i}, other-

wise:

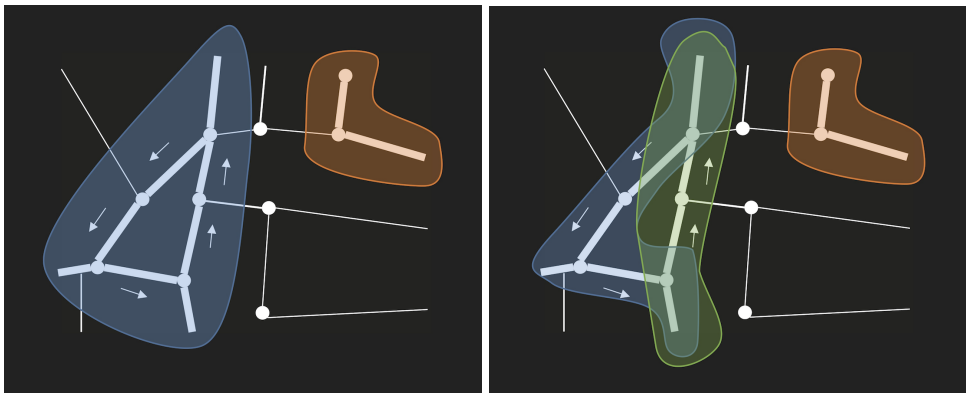
$$p(w_{i+1}|w_i) = \begin{cases} 1/k_{w_i}, & \text{if } w_i \text{ and } w_{i+1} \text{ are adjacent.} \\ 0, & \text{otherwise.} \end{cases} \tag{5.5}$$

This modification requires a more universally-applicable inference algorithm than the collapsed Gibbs sampler used for standard LDA, as introduced in the previous section. We instead use the 'universal inference engine' implemented in the STAN probabilistic programming language [20] (see Section 3.2.3 about probabilistic programming). The engine is based on a Gibbs sampler with a 'no-u-turn' extension [92] that uses adaptive parametrisation to eliminate the need of manual parameter tuning.

The proposed dependence term serves a similar purpose to the function $f(i, j)$ in the spatial community approach in (5.4). It enables spatial knowledge of the underlying street network to be accommodated while also removing the 'bag-of-words' assumption from the LDA model. The modified LDA can thus be used to detect interactional regions as collections of street segments that co-occur in vehicle journeys more often than expected based on their proximity in the underlying street network. Just like the standard LDA, the regions are represented as probabilistic distributions over street segments. Again, if we are interested in a binary partition of the street network, we assign each street segment to the region under which it has the highest probability. This time, however, the discovered regions are not influenced by spatial adjacency patterns.

5.3.4 Summary

Our methodology employs two popular clustering approaches, community detection and topic modelling, to extract interactional regions from large amounts of digital vehicle traces. At its core, both methods are designed for episodic mobility and interaction data. However, they differ in their measure of interaction between locations (Definitions 5.3.1 and 5.3.2) which leads to differences in regional delineation. Definition 5.3.1 focuses on interactions between pairs of locations and hence is better suited for origin-destination data, whereas Definition 5.3.2 looks at



(a) Interactional regions as communities **(b)** Interactional regions as topics

Figure 5.3: Interactional regions in a synthetic case in which northward and southward vehicle journeys always follow distinct routes (arrowed). White lines represent street segments, their thickness is proportional to the number of visits to a segment, i.e. the number of edges in the flow network. Since community detection (a) does not consider journeys holistically, it groups northward and southward routes into a single interactional region. On the contrary, topic modelling (b) makes a distinction between the two routes and hence discovers an additional interactional region.

interactions over sequences of locations and hence is better attuned to episodic mobility data with no functionally defined origins and destinations. When applied to mobility tracking data, the differences between the two methods are exemplified in Figure 5.3.

We attune both techniques to spatial data analysis by changing their *expected* measure of interaction between locations, i.e. before observing any vehicle traces. By default, in both methods the expected level of interaction is uniform for all pairs of street nodes. We customise it to reflect the structure of the underlying street network instead, e.g. non-adjacent street nodes are expected to have zero traffic directly between them (see Equations 5.4 and 5.5). The adaptations enable capturing interaction patterns that are not merely a result of the spatial arrangements of streets.

5.4 Numerical validation

We validate the proposed methodology on police patrol tracking data introduced in Section 2.3. Unlike other episodic activities, such as mail delivery or shopping,

Figure 5.4: Police flow network of Camden. Colour intensity of each street segment is proportional to the number of police vehicle journeys in March 2011.

police patrolling is expected to take place in every part of the neighbourhood. The granularity of spatial coverage makes them particularly suitable for validation of a regional delineation methodology such as ours. The flow network generated from the dataset is shown in Figure 5.4.

5.4.1 Interactional regions as communities

First, we extract interactional regions from police tracking data using community detection (see Definition 5.3.1). Standard community detection uncovers seventy-six interactional regions shown in Figure 5.5. The regions are small relative to the study area and are strongly clustered in space, which indicates that police patrol activities might be dominated by short-distance journeys. When we account for spatial factors according to (5.4), the resulting interactional regions (see Figure 5.6) are no longer spherical but somewhat elongated and follow stretches of individual roads.

They are also relatively small but we have no influence over region sizes when using community detection, which only outputs a single partitioning corresponding to maximal modularity in (5.2).

The differences between standard and spatial communities are intuitively plausible. Roads that are in close proximity to each other are likely to distribute high amounts of local, within region, traffic. These short-ranged interactions dominate interactional regions uncovered by standard community detection (as shown in Figure 5.5), indicating that spatial proximity plays a major role in their formation. When we incorporate spatial effects according to (5.4), we can focus on long-ranged interaction patterns instead. These rather follow long stretches of major roads (e.g. yellow community in Figure 5.6b) as high category roads attract more traffic than one would expect just based on spatial proximity.

The standard and spatial communities seem to reflect different modes of police patrolling. According to the wider literature [93] and our knowledge gathered through working closely with the Camden Police, police patrols can be roughly divided into *routine* and *emergency* patrols. The former is a form of preventive policing that require police to regularly visit crime hotspots, i.e. small geographical units with high crime intensity, such as street segments or small groups of street blocks [94]. This mode of behaviour is spatially clustered and hence well suited for standard community detection, as shown in Figure 5.5. The latter is a reactive policing effort that requires police vehicles to reach crime scenes as quickly as possible. Emergency patrolling relies heavily on major roads as they enable reaching distant crime scenes in a short amount of time. The most popular long-distance police routes are well depicted by spatial community detection in Figure 5.6. Note that these observations remain speculative, however, since there is no ground truth on what police officers actually did during their patrol journeys.

5.4.2 Interactional regions as topics

Second, we analyse interactional regions discovered using topic modelling. We begin with the standard topic modelling using LDA. Similar to standard community detection, standard topic modelling produces interactional regions which are

strongly determined by geographical factors. In contrast to community detection, however, topic modelling does not only show the *optimal* partitioning, but instead enables viewing interactional regions at multiple scales by varying the number of topics K that we fit to the data.

We show interactional regions discovered with the standard topic model at different scales in Figure 5.7. The larger they are, the more spatially constrained they become. This again reflects the fact that police journeys are predominantly local and thus short-scale interactions dominate any large-scale analysis. When very small interaction regions are chosen, standard topic modelling is capable of uncovering non-trivial interaction patterns such as long road stretches in Figure 5.7b. The ability to uncover both gravity-like and other less-trivial interaction patterns by varying the parameter K puts topic modelling at a significant advantage to community detection. In contrast to community detection, topic modelling considers complete journeys when detecting functional relations. Since journeys tend to be longitudinal, so are shapes of the extracted interactional regions. The smaller the interactional regions, the subtler the routing choices they reflect.

Topic modelling sometimes leads to disconnected parts of the street network being identified as members of the same interactional region. This rather undesirable characteristic, visible as multiple subgraphs with the same colour in Figures 5.7 and 5.8, could be addressed by a similar probabilistic model for clustering, called a block model [95]. Block model replaces the 'bag-of-words' assumption of topic modelling with a 'bag-of-pairs-of-words' assumption that places more emphasis on clustering connected parts of the network together. This could be investigated in future work.

When we switch to spatial topics and modify topic modelling to account for the underlying street network connectivity according to (5.5), we detect interactional regions with almost no spatial compactness. Even at very low resolution in Figure 5.8, they are rather stretches of roads than local neighbourhoods. The stretches often reappear in police journeys as they connect locations of mutual functional importance. In this case, they seem to be the roads connecting police stations (see

Figure 5.8b). In contrast to spatial community detection, where most interactional regions contain as few as $\sim$10 street segments (see Figure 5.9b), spatial topic modelling can uncover interactional relations at much larger distances and of generally larger sizes (Figure 5.9e). These characteristics suggest superiority of topic modelling over community detection as a method for extracting interactional regions from episodic tracking data, such as police patrol data.

5.4.3 Methods comparison

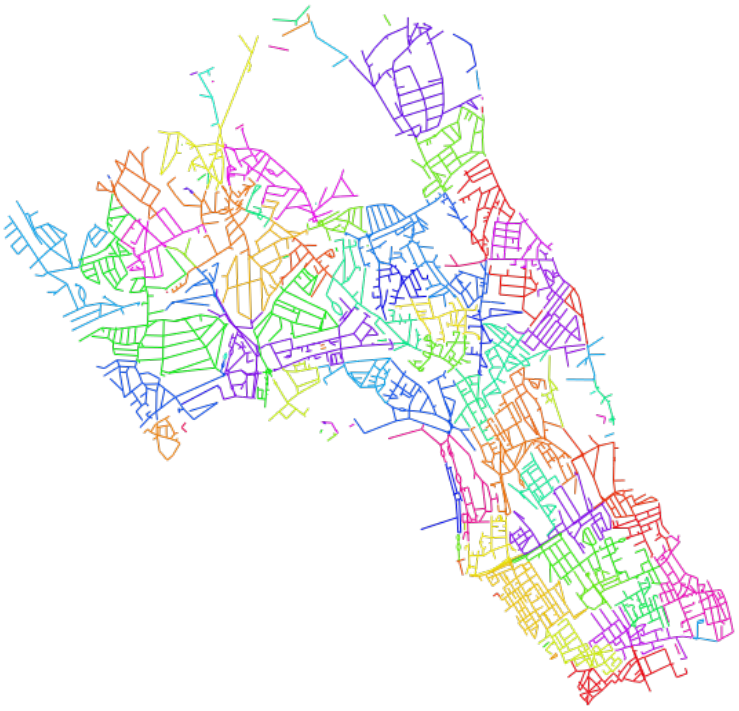
So far, our validation has focused on *qualitative* comparison of interactional regions uncovered with the proposed methods. Our focus now shifts to *quantitative* analysis to answer questions such as: how much different are the results from the different proposed methods? How can we measure their quality? We compare results from both the proposed methods, community detection and topic modelling, and their variants, standard and spatial.

We address the question of differences between regional delineations by using adjusted mutual information score (AMI) [96], which is a measure of distance between two different regional partitions. The Mutual Information (MI) is a measure of the similarity between two labels of the same data. Where $|U_i|$ is the number of samples in cluster U_i and $|V_j|$ is the number of samples in cluster V_j and N is the total number of samples, the Mutual Information (MI) between clusterings U and V is given as:

$$MI(U,V) = \sum_{i=1}^{|U|} \sum_{j=1}^{|V|} \frac{|U_i \cap V_j|}{N} \log \frac{N|U_i \cap V_j|}{|U_i||V_j|}. \tag{5.6}$$

Adjusted Mutual Information (AMI) is an adjustment of the Mutual Information (MI) score to account for chance. It accounts for the fact that the MI is generally higher for two clusterings with a larger number of clusters, regardless of whether there is actually more information shared. AMI is equal to one only when two partitions are identical and is between 0 and 1 otherwise.

Results are summarized in Figure 5.10 where we observe that interactional regions obtained from topic modelling and community detection are genuinely dif-



(a) all communities

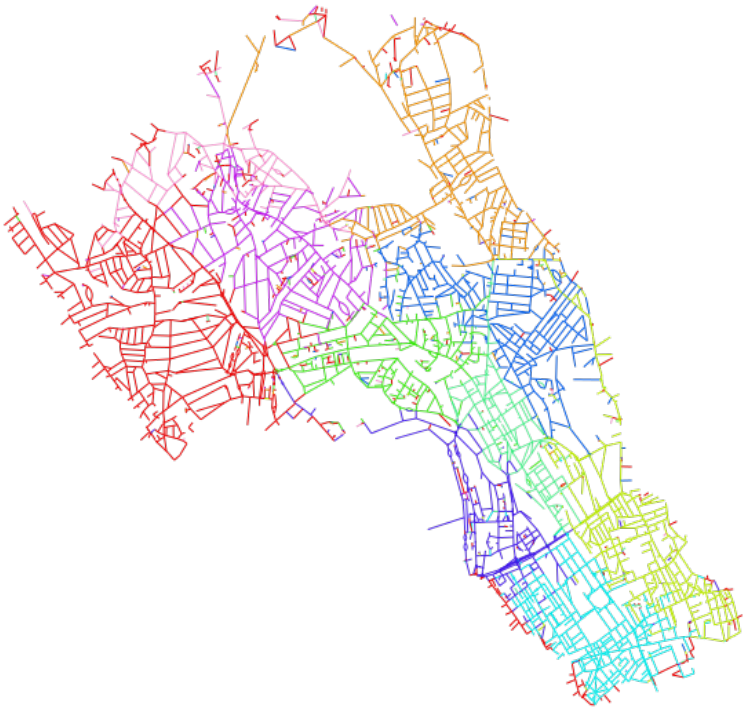
(b) largest communities

Figure 5.5: Interactional regions as communities (colour-coded).

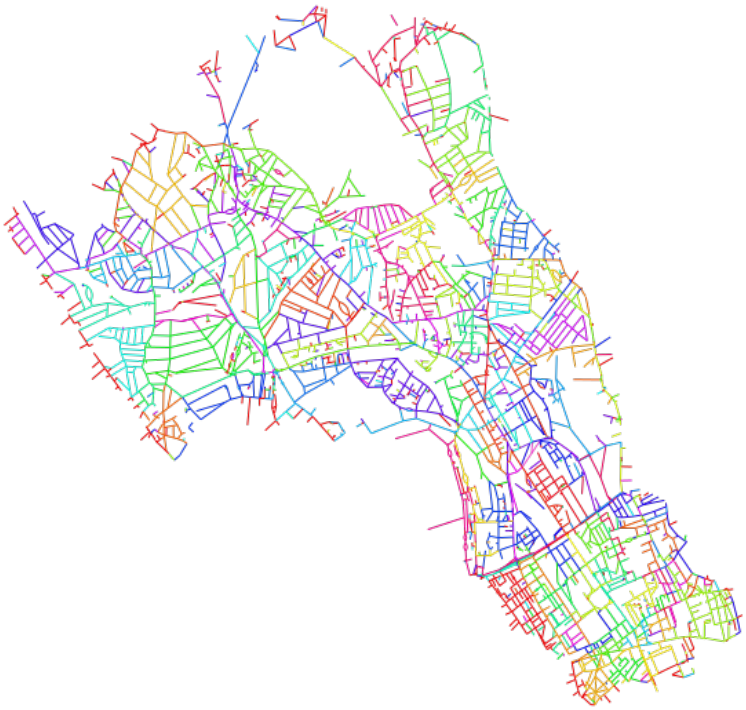
(a) all communities

(b) largest communities

Figure 5.6: Interactional regions as spatial communities (colour-coded).



(a) 10 topics



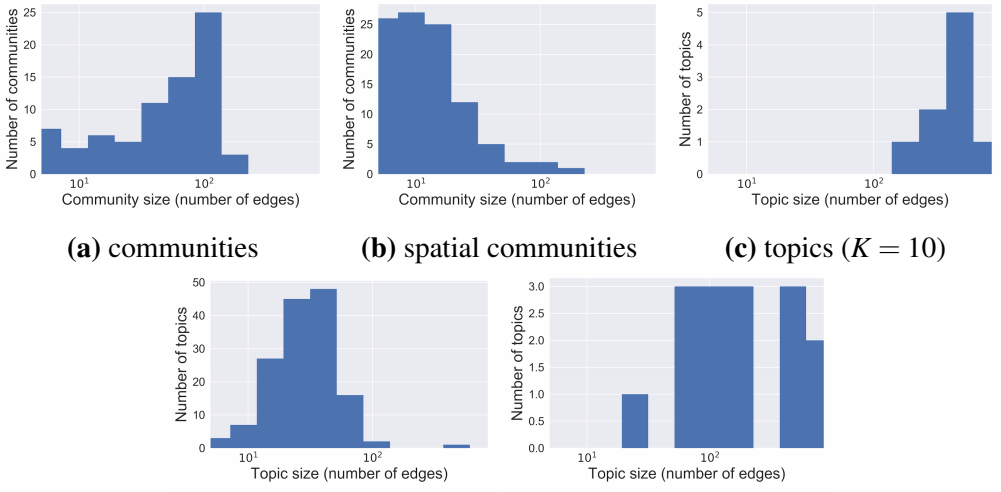
(b) 150 topics

Figure 5.7: Interactional regions as topics at different scales. Topics are colour-coded.

(a) all topics

(b) selected topics (with locations of police stations marked)

Figure 5.8: Interactional regions as spatial topics (colour-coded).



(a) communities **(b)** spatial communities **(c)** topics ($K = 10$)

(d) topics ($K = 150$) **(e)** spatial topics ($K = 15$)

Figure 5.9: Size distribution of international regions as (a) communities, (b) spatial communities, (c) 10 topics, (d) 150 topics, (e) spatial topics. Region size is equal to the number of street segments it contains. Note that region size is given in logarithmic scale on x-axis.

ferent (AMI $<= 0.35$). Their largest difference is between spatial communities and spatial topics (0.06). The dissimilarity is not surprising since the two methods follow different definitions of what interactional regions are (Definition 5.3.1 and Definition 5.3.2). On the contrary, interactional regions coming from the same method show slightly higher similarity, such as topics (K=10) and topics (K=150) (0.37). Another interesting point is that spatial methods lead to very different regional delineations to standard methods (0.35 for spatial communities vs. communities; 0.23 for spatial topics vs. topics). This difference is already acknowledged in the previous two sections, where we notice that spatial methods produce regions that are much less spatially clustered than the corresponding standard methods. By design, the difference should arise from the fact that spatial methods remove the effects of spatial proximity when looking for regional delineation.

We confirm whether the differences between spatial and non-spatial methods in fact arise from their treatment of spatial adjacency by performing a randomisation test. The test randomly shuffles the geographical position of the nodes in the flow network while keeping edges between them unchanged. As a result, in the randomised network, we observe the same volume of traffic between pairs of nodes

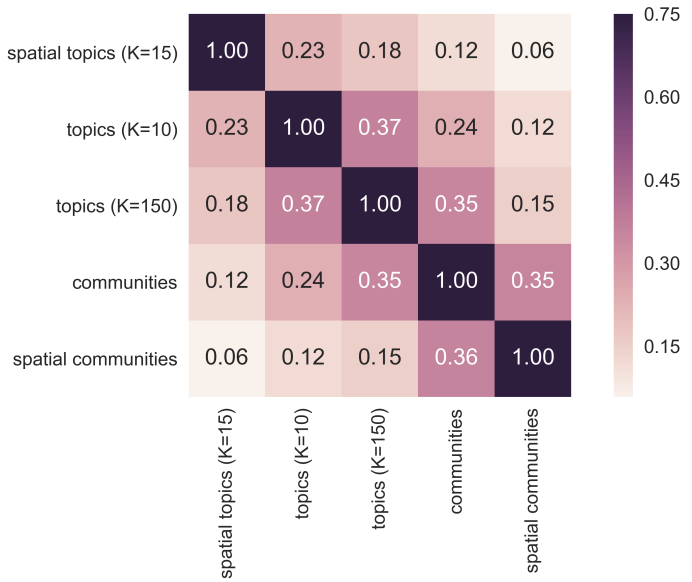


Figure 5.10: Mutual information score between regional partitions obtained with the proposed methods.

	Original-Random
Spatial Topics ($K = 15$)	0.01613398 ± 0.02
Spatial Communities	0.35651846 ± 0.03

Table 5.2: Average MI measured between the regional partition found on the original flow network and 100 randomized networks (Original-Random) for spatial topic modelling and spatial community detection.

but the traffic can now occur between nodes that are not connected by a street segment in reality (thus violating our spatial adjacency assumptions in (5.4) and (5.5)). The randomised network is no longer embedded in the underlying street network, but this has no effect on regional partitions uncovered with *standard* community detection and topic modelling, since these methods do not make use of the street arrangement information. On the contrary, *spatial* community detection and topic modelling, which assume the spatial embedding of the network, uncover partitions that are largely different from the ones they find in the real, spatially-embedded, flow network (see Table 5.2). Interestingly, the largest variation is shown by spatial topic modelling, where partitions found on the random and the real network show almost zero resemblance (0.016).

The randomisation test could be further extended to measure significance of the discovered partitions as in the work by [79]. This would bring us close to answering the second question posed: can we measure quality of the discovered partitions? The extension could constitute future work but it would, nonetheless, not fully answer the question of partition quality, which is conceptually difficult to answer in the lack of ground truth on the correct interactional region delineation. If ground truth was available, we could assess the quality of different partitions by measuring their similarity to the ground truth using the mutual information score, as shown for different partitions in Figure 5.10. For example, [79] used two different linguistic communities in Belgium as ground truth partitions for a spatially-embedded network of phone calls. Since we are not in possession of such ground truth, we can only approximate partition quality through significance testing (as suggested above) or through usefulness of the discovered partition in a specific context, e.g. ”do the discovered regions lead to increased accuracy in region-based route choice simulations?” as explored in the work by [51], or ”are the discovered partitions more stable across time?”. Answers to these questions could create interesting extensions to this chapter but are outside the scope of this work.

5.5 Discussion and directions for future research

So far in this chapter, we have presented a comprehensive network-based methodology for extracting interactional regions from digitised vehicle traces in urban environments. The methodology used a large dataset of GPS vehicle traces to define a road traffic network and then uncovered interactional regions as densely connected areas within the network. It considered two approaches to the discovery of interactional regions: community detection and topic modelling. Community detection used aggregated traffic flows between pairs of street nodes when assigning them to regions. Topic modelling instead considered sequences of street nodes corresponding to complete vehicle journeys, hence potentially giving a more complete picture of how drivers perceive the urban space. The techniques were adapted to account for the effect of space upon the network topology, hence uncovering interaction

patterns between places that arose not solely from spatial proximity.

Both community detection and topic modelling could detect short-ranged interaction patterns in police patrol data, suggesting that spatial proximity is a major force in spatial interactions. By adopting the methods to focus on spatially-anomalous interactions only, they could also uncover less-trivial long-ranged interaction patterns. Therefore, the proposed methodology, including both standard and spatial adaptations of the methods, provides a more detailed insight into interaction patterns than previously proposed using standard community detection only [72, 77, 73]. Our initial analysis suggested two advantages of topic modelling over community detection for episodic activity data. Firstly, it could detect either aggregate or granular activity patterns by varying a single parameter K. Secondly, it was capable of detecting longer activity trails, such as paths between police stations. On the negative side, however, topic modelling could lead to disconnected parts of the street network being classified as members of the same interactional region. This might be undesirable when designing effective police districts or administrative boundaries.

We validated our methodology using police patrol data due to their availability and high spatial granularity. We discovered interactional regions which seemed to correspond to two modes of police patrols: routine and emergency patrols. Spatially clustered interactional regions bounded routine patrol activities, restricted to neighbourhoods and minor roads, whereas elongated interactional regions corresponded to popular major road routes leading to police stations or emergency calls. We uncovered distinguishable patrolling preferences that could potentially be used in the design of effective police districts, especially in light of recent funding cuts and multiple London police districts being merged.

Our comprehensive methodology extends beyond police patrol data. It is applicable to vehicle flows in general, as well as other episodic flows, e.g. cyclist or pedestrian journeys. As such, the methodology opens new avenues of quantitative analysis of urban dynamics. Depending on the dataset analysed, it can discover regional partitions that are case study-specific (e.g. regions extracted from *police*

journeys) or more generalizable to the wider city population (e.g. regions extracted from *all* vehicle journeys).

Future research could extend the proposed methods to account for subtler spatial effects, such as differences in flows on major and minor roads. It could also validate the methods further using other city flow data or by using the discovered regions within a specific application, such as region-based route choice simulation frameworks (see the next section as an example).

5.6 Application: region-based route prediction

So far in this chapter, we have presented two approaches to regional delineation from vehicle tracking data: topic modelling and community detection. In this section, we are particularly interested in the topic modelling approach because it derives regions directly from our intuitions, grounded in route choice theory [97, 98, 52] and encoded as a probabilistic graphical model, that people plan their journey to a new destination as a sequence of preferred routes through different neighbourhoods. Drivers' behaviour is shown to be suboptimal and their "route selection takes place in phases, linking locations and decision points on route to destination" [98]. We can use the extracted regions as building blocks when designing routes to new destinations.

This section shows preliminary experiments of using topics, obtained using topic modelling, to predict route choices of drivers in cities. Although limited previous research used topic models to discover mobile behavioural patterns [99, 100], no research has adopted the discovered patterns for movement modelling. Therefore, the work presented in this report, despite its premature state, could potentially contribute to the research community not only by applying topic modelling to a new type of data, but also through an innovative use of the discovered topics in vehicle route choice modelling. We validate the proposed methodology on routes taken by police patrol vehicles in the London Borough of Camden (as introduced in Section 2.3).

5.6.1 Methodology

5.6.1.1 Route choice modelling using topics

As in Section 5.3.3.1, topics are inferred from vehicle GPS traces using Latent Dirichlet Allocation (LDA) with the collapsed Gibbs sampler as the inference engine. Since the Gibbs sampler is tailored specifically to LDA, there is no need to resort to more universally applicable, but often less efficient, inference engines offered by probabilistic programming languages such as STAN (see Section 3.2.3 about probabilistic programming).

The topics are inferred as distributions over street segments. They reveal underlying clusters of street segments based on the segments' co-appearance in vehicle journeys and hence enable a simplified representation of vehicle journeys as distributions over the topics.

The topics are then used to model police vehicle movement. The original street network is reduced to a topic network, in which each topic node is a collection of street segments that have the highest probability of appearing in that topic. Topic nodes are connected by an edge if their street segments are physically connected. Vehicles that want to travel from one street segment to another start at the topic node where the first segment is assigned and take the shortest path through the topic network from that node to the topic node containing their journey destination. This modelling framework reflects the intuition that journeys are undertaken in stages or as series of topics on route to the destination.

Two variants of the modelling framework are explored. In the first one, the topic network is unweighted and vehicles choose paths that minimise the number of topics traversed to their destination. In the second one, topic nodes are weighted by the total length (in metres) of street segments assigned to them. Vehicles subsequently choose paths that minimise journey length.

5.6.1.2 Model validation

The models are validated against the police tracking data by measuring the Pearson correlation coefficient [101] between street coverage generated by the models

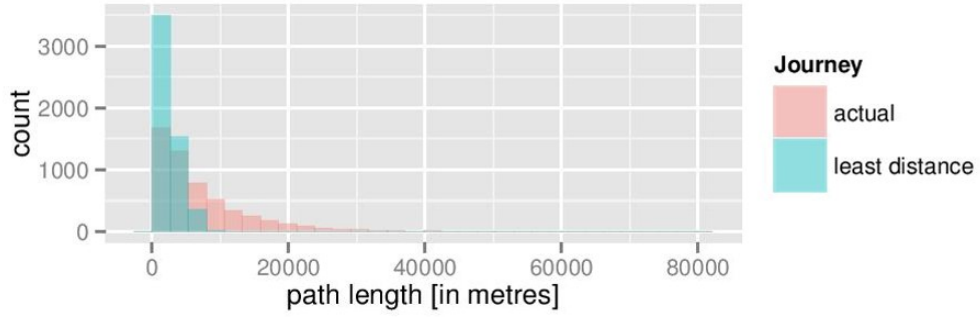


Figure 5.11: Distributions of lengths of journeys in March 2010 and their least distance alternatives.

and the actual street coverage, where coverage is defined as the number of vehicle visits at each street segment in Camden. The generated street coverage contains journeys between all possible origins and destinations in the Camden's street network, weighted by the probability of observing such an origin-destination pair in the actual data.

Since each generated journey has an observation probability, the coverage at a street segment is represented by the sum of probabilities of all possible journeys that would visit that segment. An alternative approach would be to simulate agents (vehicles) based on the probabilistic rules and then use their journeys to calculate the generated coverage. The alternative approach would be prone to sampling bias though, especially if the number of agents was small.

5.6.2 Results and discussion

5.6.2.1 Police vehicle data

Police vehicle data motivating the project are introduced in Section 2.3 and depicted in Figure 2.1. Route choice preferences observed in the data are sub-optimal in terms of journey length as shown in Figure 5.11 and strongly biased towards the use of major roads (Figure 5.12). These patterns are in line with the findings by [98] that underpin our models.

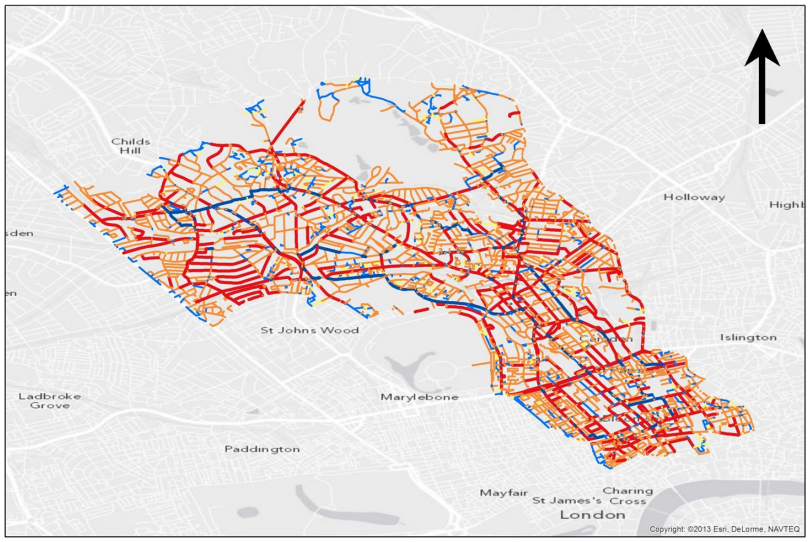


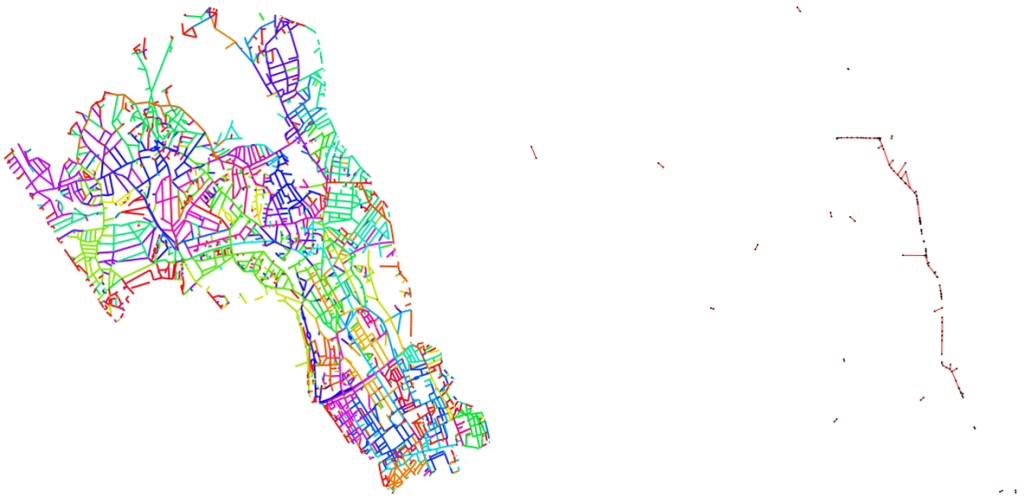
Figure 5.12: Difference between actual and least distance journeys (*actual* minus *least distance*) in March 2010; yellow corresponds to exact match.

5.6.2.2 Route choice modelling

Topics inferred from the data are shown in Figure 5.13. Latent Dirichlet allocation algorithm required specifying the number of topics a priori and this number was set to hundred following initial experiments into the influence of the number of topics on topic sizes. However, it is acknowledged that further research is required into optimising the number of topics for modelling purposes. For example, the number of topics could be tuned to maximise the accuracy of route choice simulations, as measured using the Pearson correlation coefficient between the generated and the actual street coverage.

Figure 5.13 shows street segments coloured by topics under which they have the highest probability. The colours seem to reflect the network proximity, as well as a general road hierarchy. Segments of major roads tend to be clustered together forming 'stretched' topics, whereas segments of minor roads seem to be clustered within their neighbourhoods. These observations reflect the intuition that vehicles tend to travel longer distances along major roads but otherwise limit their journeys to nearby locations.

The inferred topics are used to create a topic network. This proves to be prob-



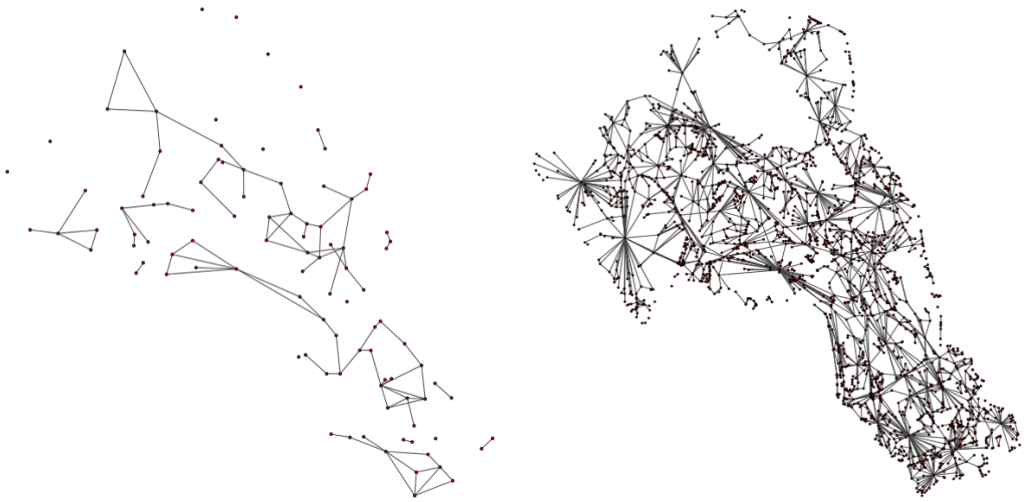
(a) (b)

Figure 5.13: (a) Camden's street network coloured by topic assignments, (b) an example topic, when hundred topics are inferred from police journeys in March 2010 in the London Borough of Camden. Topic assignments are obtained by colouring each street segment by the topic under which they have the highest probability.

lematic as a closer investigation of topics in Figure 5.13 reveals that topics are often disconnected (see example topic in Figure 5.13b). The discontinuity might be due to our simplistic approach to assigning segments to topics, in which a topic is treated as a defined collection of street segments rather than a distribution over all street segments. The issue requires further investigation though, which is outside the scope of this research.

For modelling purposes, the discontinuity is tackled by creating a topic graph in which each topic is represented by multiple nodes, each representing one of its connected components. The resulting topic graph is shown in Figure 5.14b. An alternative approach could construct the graph based on the largest connected component of each topic only. This would, however, lead to a disconnected graph as shown in Figure 5.14a.

The topic graph is used to model police vehicle movement according to the procedures introduced in Section 5.6.1.1. An example journey generated according to the procedures is shown in Figure 5.15. Across all journeys, the generated coverages for the unweighted and weighted model variants are shown in Figure 5.16. Their correlations with the actual police coverage are **0.204** and **0.349** respectively.



(a) **(b)**

Figure 5.14: Topic graph created from (a) only the largest connected component of each topic, (b) all connected components of each topic, when hundred topics are inferred from police journeys in March 2010.

Figure 5.15: An example journey generated as the shortest path the topic graph (right), also shown on the underlying street network (left).

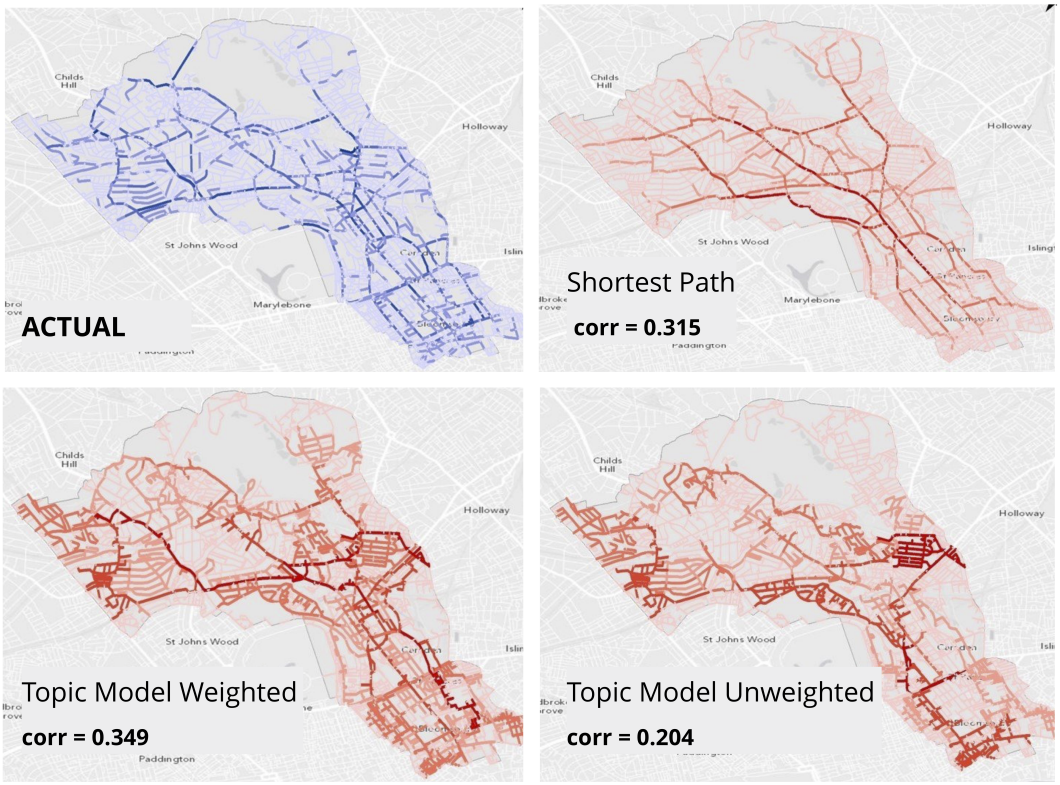


Figure 5.16: Actual police coverage (top left) versus police coverage generated using the least distance assumption (top right), unweighted (bottom left) and weighted (bottom right) topic graph from Figure 5.15. Colour intensity is proportional to the number of journeys passing through each street segment.

Although these correlations are not substantially higher than a correlation of 0.315 between the data and a simplistic model assuming that vehicles always follow least distance paths, coverage patterns that they generate reflect subtle route choice preferences visible in the actual police coverage that cannot be captured under the unrealistic least distance assumption. For example, the proposed models reconstruct high volumes of traffic observed in the western- and southernmost borders of the Camden borough (see Figure 5.16).

Further work is required to uncover the full potential of using topics in movement modelling. Possible extensions of the work presented in this scenario include:

- automated inference of the number of topics that would maximise modelling accuracy,
- higher order correlation metrics to measure model accuracy (e.g. correlation

between counts on adjacent segments could better reflect the accuracy of a model in reconstructing movement patterns),

- addition of network connectivity information to the topic modelling algorithm in order to increase connectedness of discovered topics.

5.7 Conclusions

This chapter presents a comprehensive toolkit for region extraction from digitised vehicle traces. The regions are reflective of people's routing preferences and as such could be used in the design of effective urban districts or in simulating realistic drivers' behaviour.

The chapter introduces topic modelling as a model-based approach to regional clustering. Topic modelling is capable of clustering streets based on their co-occurrence in long vehicle journeys, which makes it particularly well suited for regional delineation from such episodic activities. In contrast to community detection, topic modelling can uncover regions at different levels of granularity by varying a single parameter K. On the downside, topic modelling can uncover regions that are spatially disconnected. Further work could address this limitation by replacing the 'bag-of-words' assumption of topic modelling with a 'bag-of-pairs-of-words' assumption of a related block model [95], which would place more emphasis on clustering connected parts of the network together.

The chapter extends both topic modelling and community detection for uncovering space-independent patterns in the traffic network. By factoring out the effect of space, the methods discover regions corresponding to long-distance routes between points of interest. Future work should focus on further validation of the methods. One direction could use randomisation testing to measure significance of the discovered partitions as in the work by [79]. Other direction could test the methods in a controlled setting by generating synthetic spatially-embedded networks with known regional partitions. The methods could then be compared on their ability to uncover regional partitions corresponding to spatial and non-spatial forces.

Finally, the chapter uses the discovered regions for route choice prediction.

Initial results show improved accuracy in comparison to route prediction using the shortest-path assumption. Further work is needed to develop higher order correlation metrics for method evaluation and to tune the size of regions used for route simulation.

Chapter 6

Methodological extensions

Driven by questions on urban dynamics, we advance the current state-of-the-art in Bayesian inference in probabilistic graphical models. In this chapter, we present our most fundamental work in Bayesian inference that can benefit a far broader range of applications than the initial domain of urban dynamics.

Our methodological work has started off with the question of ”where you are” in cities, inferring true positions from noisy tracking data, which requires sampling possible locations on highly complex road networks. We usually represent possible positions with a distribution that we know how to sample from, such as a Gaussian distribution in Figure 4.16. But what if we could sample from more realistic, complex distributions over road networks as exemplified in Figure 6.1? Inspired by the question, our research proposes a more fundamental contribution to Bayesian inference with complex, non-linear posterior distributions.

The chapter is split into two parts. Firstly, we propose a novel finite-sample inference algorithm, called adversarial sequential Monte Carlo, which is capable of approximating highly non-linear posterior distributions. The algorithm uses a deep generative network as its proposal distribution that is adversarially trained to approximate the desired posterior. The idea contributes to recent research into Bayesian deep learning, bringing together ideas from probabilistic machine learning and generative adversarial networks (GANs) in order to design a next generation Bayesian inference algorithm. The algorithm suffers from training instability during proposal training, which we address with the ongoing research below.

Figure 6.1: Candidate positions for each GPS observation could be sampled from a realistic complex distribution (shaded in blue) around each observation which we could learn from data.

Secondly, we propose an improved approach to training deep generative networks, which utilizes a technique from statistical hypothesis testing known as maximum mean discrepancy (MMD). The approach leads to a much simpler training objective than the two-player objective of GANs [102]. Initial results indicate that deep generative networks trained using MMD distance train faster and achieve higher accuracy than equivalent networks adversarially trained. We train deep generative networks using MMD distance in order to sample latent codes given observations. Future work will apply the trained networks as proposal models with a sequential Monte Carlo inference algorithm, such as particle filter.

6.1 Adversarial particle filters

6.1.1 Overview of our method

How can we perform efficient inference in directed probabilistic models with intractable posterior distributions? We introduce a new technique for improving finite-sample inference approximations by learning highly flexible proposal distributions for sequential importance samplers, such as particle filters. We represent proposal distributions as implicit generative models, that is models that you can sample from but which do not have an explicit parametric form, and train them using variational inference rephrased as a two-player game, hence establishing a principled connection between Sequential Monte Carlo (SMC) and Generative Adversarial Networks (GANs). Our approach achieves state-of-the-art performance on synthetic and real

inference problems.

6.1.2 Introduction

Sequential Monte Carlo, or particle filtering, is a popular class of methods for Bayesian inference that approximate an intractable target distribution by drawing samples from a series of simpler intermediate distributions. SMC methods have traditionally been used for filtering in state-space models [28, 42], but have since shown state-of-the-art results in a far broader range of models, including factor graphs, hierarchical Bayesian models [103] and general probabilistic programs [24, 104].

Critical to the performance of SMC is the choice of appropriate proposal distributions for transitioning from one intermediate distribution to the next. Theoretically optimal proposal distributions [27] are in general intractable, thus in practice they are often approximated using a prior distribution, as in the case of bootstrap particle filter [28]. More adaptable proposal distributions have recently been proposed by [105, 106] who use neural networks with parametrised distributions as output layers to design distributions of varying complexity. Their quality is largely dependent on the suitability of their parametrisation to the problem at hand, however, as well as the availability of large volumes of training data. All in all, the ability to automatically learn appropriate proposal distributions for any inference problem is still a real need that we need to address before SMC methods can be automatically applied to new models and problems.

In this project, we investigate the use of Generative Adversarial Networks (GANs) [102] for constructing highly flexible proposal distributions for Sequential Monte Carlo. GANs provide a tool to learn implicit proposal models from data, i.e. models that define a stochastic procedure that directly generates proposal samples [107, 108], which in contrast to parametric proposal models [105, 106], can learn proposal models of arbitrarily complexity. The proposals are black-box sample generators that are adversarially trained to obtain a close approximation to a target posterior distribution. They offer high flexibility and remove the need to design a suitable parametrisation for a new inference problem [109], hence advancing

our work towards automatic SMC inference.

Similar previous work explored the use of GANs for training arbitrarily expressive inference models for Variational Autoencoders [109] and has shown empirically compelling results as well as theoretical guarantees of the convergence of the inference model to the true posterior distribution over the latent variables given an observation. In this work, we explore the use of GANs for training arbitrarily complex proposal models for sequential Monte Carlo in order to accelerate it across a diverse range of problem settings. Our contributions are as follows:

- We enable learning appropriate proposal models for SMC using adversarial training.
- We give theoretical insights into our method, deriving adversarial proposal learning from the principles of Kullback-Leibler (KL) divergence minimisation.
- We empirically demonstrate that our proposal models lead to improved SMC performance in both synthetic and real case studies.

6.1.3 Methodology

Sequential Monte Carlo, introduced in Section 3.3.3.1, is an importance sampling scheme that approximates a posterior distribution $p(x_{1:N}|y_{1:N})$ over N-dimensional latent variable $x_{1:N}$ by a set of weighted samples from a (presumably simpler) proposal distribution $q(x_{1:N}|y_{1:N})$. SMC methods perform best when the proposal distribution is close the intractable posterior distribution. One direction for innovation in SMC algorithms, which this work contributes to, involves proposing novel methods for constructing proposal models that approximate the target posterior as close as possible.

In this work, we show how we can represent highly flexible proposal distributions $q(x_{1:N}|y_{1:N})$ for sequential importance samples as implicit generative models and use adversarial training to obtain a close approximation $q^*(x_{1:N}|y_{1:N})$ to the true posterior $p(x_{1:N}|y_{1:N})$. After training, the proposals take as input particular values of the observed random variables, and return an approximation to the distribution of

the latent variables. The proposals can be used to accelerate sequential Monte Carlo inference across a diverse range of problems. Our method is shown schematically in Figure 6.2.

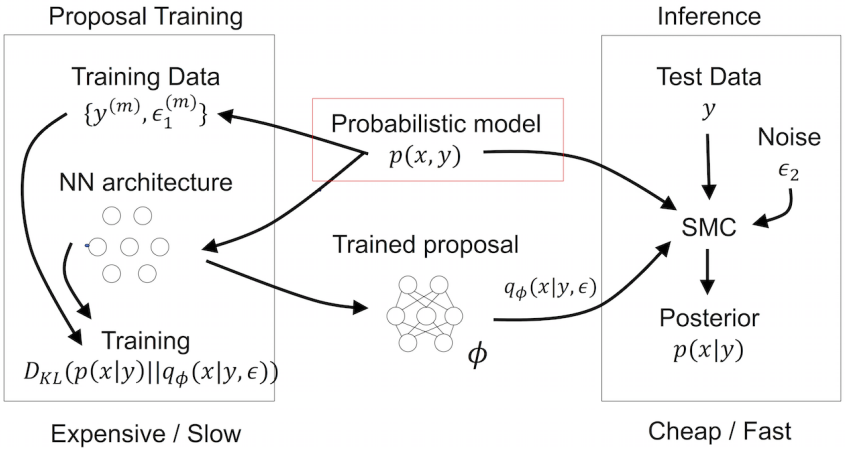


Figure 6.2: Our approach to sequential Monte Carlo with implicit proposal models. We represent a proposal model as a fully connected deep network architecture that accepts samples from a noise variable ε_1 and an observed variable y and outputs samples of a latent variable x. We then train the network given a stream of observed data $\{y^{(m)}\}$ and the corresponding generative model $p(x,y)$. Once this expensive training stage is complete, we are left with an artifact of weights ϕ and neural network architecture specialised for the given generative model. During *inference*, the trained artifact is used as a proposal model within a sequential Monte Carlo sampler.

6.1.3.1 Objective for proposal learning:

Our goal is to find a proposal distribution $q_\phi(x_{1:N}|y_{1:N})$ parametrised by ϕ that closely approximates the target posterior distribution $p(x_{1:N}|y_{1:N})$. We frame the problem as variational inference (see Section 3.2.2.1), that is, in terms of finding proposal parameters ϕ^* that minimise the KL divergence between the proposal and the target densities

$$\begin{aligned}
&\min_\phi KL[q_\phi(x_{1:N}|y_{1:N})||p(x_{1:N}|y_{1:N})] = \\
&= \min_\phi E_{q_\phi} \log \frac{q_\phi(x_{1:N}|y_{1:N})}{p(x_{1:N}|y_{1:N})} \\
&= \min_\phi E_{q_\phi} [\log \frac{q_\phi(x_{1:N}|y_{1:N})}{p(x_{1:N})} - \log p(y_{1:N}|x_{1:N})] + \log p(y_{1:N})).
\end{aligned} \tag{6.1}$$

We ignore the marginal likelihood term $\log p(y_{1:N})$, since it does not depend on ϕ, and use the factorisation of our joint distribution in (3.16) to arrive at our final optimisation objective for $n = 1,..,N$:

$$\min_{\phi_n} E_{q_{\phi_n}}\left[\log \frac{q_{\phi_n}(x_n|x_{1:n-1},y_{1:n})}{p(x_n|x_{1:n-1})} - \log p(y_n|x_{1:n},y_{1:n-1})\right] \tag{6.2}$$

where q_{ϕ_n} are intermediate proposals as in (3.19) parametrised by ϕ_n.

The final optimisation objective in (6.2) shows that we can break down the problem of learning the full proposal density $q_\phi(x_{1:N}|y_{1:N})$ into N problems of learning intermediate densities $q_{\phi_n}(x_n|x_{1:n-1},y_{1:n})$ and we are still guaranteed to satisfy the original optimisation objective in (6.1).

We can exploit the factorisation to reduce the dimensionality of the optimisation problem for models where hidden and observed variables satisfy the *Markov property*, that is, where $p(x_n|x_{1:n-1}) = p(x_n|x_{n-1})$ and $p(y_n|x_{1:n},y_{1:n-1}) = p(y_n|x_n)$:

$$\min_{\phi_n} E_{q_{\phi_n}}\left[\log \frac{q_{\phi_n}(x_n|x_{n-1},y_n)}{p(x_n|x_{n-1})} - \log p(y_n|x_n)\right] \tag{6.3}$$

and even more for models that are also *time-invariant*, that is, where $p(x_n|x_{n-1})$ and $p(y_n|x_n)$ are independent of n, as they only require a single set of parameters ϕ to be learnt:

$$\min_{\phi} E_{q_{\phi}}\left[\log \frac{q_{\phi}(x_n|x_{n-1},y_n)}{p(x_n|x_{n-1})} - \log p(y_n|x_n)\right] \tag{6.4}$$

6.1.3.2 Adversarial learning:

Now we describe the procedure for training each factor q_{ϕ_n} of the proposal distribution according to the optimisation objective in (6.4). Contrary to many standard settings in which one is limited by the amount of data present, we are in possession of the generative model $p(x,y)$ which allows us to sample effectively infinite training sequences of latent and observed variables $\{x_{1:n},y_{1:n}\}$. In practice, we found that our method performed well when trained using relatively small data sets.

We represent each factor q_{ϕ_n} of the full proposal as an *implicit probabilistic model*, that is a model that specifies a stochastic procedure that directly generates a

latent variable x_n [107]. Implicit proposal models use an input noise variable $g(\varepsilon)$ and a deep network architecture x_{ϕ_n} parametrised by ϕ_n to flexibly characterise a wide range of proposal distributions:

$$x_n = x_{\phi_n}(x_{1:n-1}, y_{1:n}, \varepsilon'), \qquad \varepsilon' \sim g(\varepsilon) \tag{6.5}$$

Implicit formulation requires adversarial training to find ϕ_n that optimise our learning objective in (6.2). We introduce a discriminator network T with the following objective

$$\max_T E_{q_{\phi_n}(x_n|x_{1:n-1},y_{1:n})} \log \sigma(T(x_{1:n}, y_{1:n})) + E_{p(x_n|x_{1:n-1})} \log(1 - \sigma(T(x_{1:n}, y_{1:n}))) \tag{6.6}$$

where σ denotes the sigmoid function. Intuitively, T tries to distinguish between pairs $(x_n, y_{1:n})$ with x_n sampled using the prior $p(x_n|x_{1:n-1})$ from those sampled using our proposal $q_{\phi_n}(x_n|x_{1:n-1}, y_{1:n})$, where $y_{1:n}$ are observations. The optimal discriminator T^* is given by

$$T^*(x_{1:n}, y_{1:n}) = \log \frac{q_{\phi_n}(x_n|x_{1:n-1}, y_{1:n})}{p(x_n|x_{1:n-1})}. \tag{6.7}$$

Hence, we can use (6.5) and (6.7) to write the optimisation objective in (6.2) as

$$\min_{\phi_n} E_\varepsilon (T^*(x_{1:n-1}, x_{\phi_n}, y_{1:n}) - \log p(y_n|x_{1:n-1}, x_{\phi_n}, y_{1:n-1})). \tag{6.8}$$

Here, we parameterize the neural network T with a vector ψ. We adapt ϕ_n and ψ in parallel to optimise our objective for proposal learning in (6.2) by alternating gradient calculations for the discriminator according to (6.6) treating ϕ_n as fixed and gradients for (6.8) treating ψ as fixed. In all experiments, we train using Adam optimiser [110] and improve the robustness of training using Adaptive Contrast technique recently proposed by [109].

Algorithm 3 Adversarial proposal training.

$i \leftarrow 0$;
while not converged **do**
  Sample $\{y_{1:n}^{(1)}, ..., y_{1:n}^{(m)}\}$ from data distrib. $p_D(y_{1:n})$;
  Sample $\{x_n^{(1)}, ..., x_n^{(m)}\}$ from prior $p(x_n|x_{1:n-1})$;
  Sample $\{\varepsilon^{(1)}, ..., \varepsilon^{(m)}\}$ from $\mathcal{N}(0,1)$;
  Compute ϕ-gradient (eq. 6.8):
  $g_\phi \leftarrow \frac{1}{m}\sum_{l=1}^{m} \nabla_\phi \big[T_\psi(x_{1:n-1}^{(l)}, x_{\phi_n}(x_{1:n-1}^{(l)}, y_{1:n}^{(l)}, \varepsilon^{(l)}), y_{1:n}^{(l)})$
    $-\log p(y_n^{(l)}|x_{1:n-1}^{(l)}, x_{\phi_n}(x_{1:n-1}^{(l)}, y_{1:n}^{(l)}, \varepsilon^{(l)}), y_{1:n-1}^{(l)})\big]$;
  Compute ψ-gradient (eq. 6.6):
  $g_\psi \leftarrow \frac{1}{m}\sum_{l=1}^{m} \nabla_\psi \big[\log(\sigma(T_\psi(x_{1:n-1}^{(l)}, x_{\phi_n}(x_{1:n-1}^{(l)}, y_{1:n}^{(l)}, \varepsilon^{(l)}), y_{1:n}^{(l)})))$
    $+\log(1-\sigma(T_\psi(x_{1:n}^{(l)}, y_{1:n}^{(l)})))\big]$;
  Perform SGD-updates for ϕ and ψ:
  $\phi \leftarrow \phi - h_i g_\phi, \qquad \psi \leftarrow \psi + h_i g_\psi$;
  $i \leftarrow i+1$;
end while

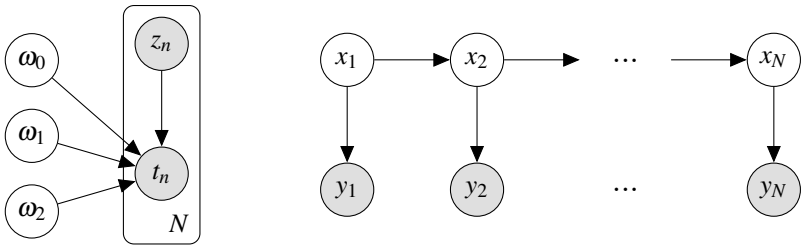


Figure 6.3: Directed graphical model for (left) non-conjugate regression and (right) first-order Markov chain.

6.1.3.3 Adversarial sequential Monte Carlo:

We find optimal ϕ_n off-line for $n = 1, ..., N$ and then use the trained proposal models q_{ϕ_n} within an SMC algorithm called particle filters [111]. Since the proposal distributions are represented as implicit models, that is, models that we can sample from but whose density we cannot evaluate, we assume a uniform density across all their samples. This is a very simplistic assumption that will be addressed in future work. The assumption simplifies unnormalised particle weights in (3.20) to:

$$w(x_{1:N}) = w_1(x_1) \prod_{n=2}^{N} p(x_n|x_{1:n-1}) p(y_n|x_{1:n}, y_{1:n-1}). \tag{6.9}$$

6.1.4 Performance evaluation

We use a number of metrics to evaluate the performance of our method. In experiments where ground truth states $x_{1:N}$ are known, we measure the root mean square error (RMSE) between the approximate posterior given by the mean of K particles $\tilde{x}_{1:N}$ and the true states

$$RMSE(x_{1:N}, \tilde{x}_{1:N}) = \Big(\frac{1}{N}\sum_{n=1}^{N}(x_n - \tilde{x}_n)^2\Big)^{(1/2)}. \tag{6.10}$$

More generally, we use the estimate of the marginal likelihood in (3.25) to evaluate the log-marginal likelihood (LML) given observed sequence $y_{1:N}$

$$\begin{aligned} LML = \log(p(y_{1:N})) &= \sum_{n=1}^{N} \log(p(y_n|y_{1:n-1})) \\ &= \sum_{n} \log\Big(\frac{1}{K}\sum_{k=1}^{K} w_n^{(k)}\Big) \end{aligned} \tag{6.11}$$

and calculate a common metric for assessing performance of SMC methods called the effective sample size (ESS). ESS of particles at time n is given by

$$ESS_n = \Big(\sum_{k=1}^{K}(W_n^{(k)})^2\Big)^{-1}. \tag{6.12}$$

where $W_n^{(k)}$ are the *normalised* importance weights in (3.23). ESS is maximised and equals the number of particles K when all particles are assigned uniform importance weights. Note that ESS measures the relative quality of the samples rather than their absolute quality, so ESS should not be used as the only performance metric.

6.1.5 Experiments

6.1.5.1 A linear regression

We begin by illustrating our method with a non-conjugate polynomial regression model proposed by [105], with global-only latent variables. The model is shown in Figure 6.3 (a). The model defines a Laplace prior on the weights and Student-t

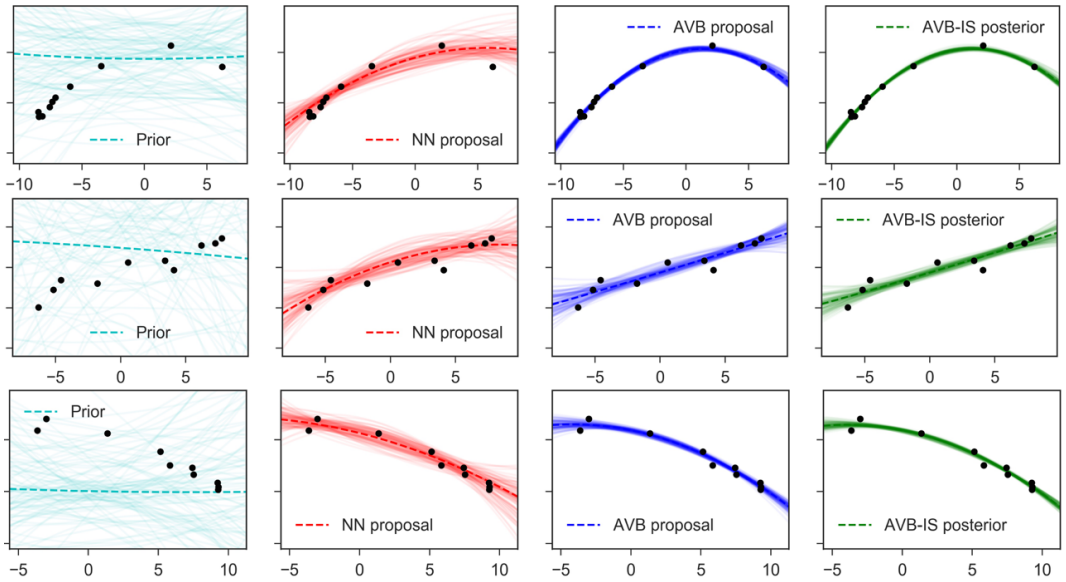


Figure 6.4: Example output in the polynomial regression case study. Plots show 100 regression curves, each curve estimated from one sample of weights, and the mean curve marked as a dashed line. In the leftmost column, weights are sampled from the Laplace prior. Our proposal (AVB) and posterior after importance sampling (AVB-IS) yield estimates close to proposal by [105] (NN), but slightly less diffused. Black dots represent true observations corresponding to different true weights in each row.

likelihoods, giving us

$$\begin{aligned} p(\omega_d) &= Laplace(0, 10^{1-d}) \qquad \text{for } d = 0,1,2 \\ p(t_n|\omega_0,\omega_1,\omega_2,z_n) &= t_v(\omega_0 + \omega_1 z_n + \omega_2 z_n^2, \varepsilon^2) \end{aligned} \tag{6.13}$$

for $n = 1,...,N$, fixed $v = 4$, fixed $\varepsilon = 1$ and $z_n \in (-10,10)$ uniformly. Our goal is to infer latent variables $x \equiv \{\omega_0,\omega_1,\omega_2\}$ given observed variables $y \equiv \{z_n, y_n\}_{n=1}^N$. Since our latent variables are independent of time $n = 1,...,N$, we train a single proposal distribution $q_\phi(\omega_{0:2}|z_{1:N}, y_{1:N})$ that can simultaneously sample all latent variables given a sequence of observations. We arrive at samples approximating the posterior $p(\omega_{0:2}|z_{1:N}, y_{1:N})$ by applying a single step of importance sampling to samples from the proposal distribution.

We represent our proposal by a two-layer network with 128 hidden units in each layer. As in the work of [109], the network accepts the noise ε as additional input to *implicitly* model the proposal distribution. We train the proposal against a

slightly more powerful discriminator represented by a five-layer network with 256 hidden units in each layer. We show example samples from the proposal and the posterior after importance sampling in Figure 6.4. Alongside, we show samples from a proposal given by the same network, but with a mixture of three Gaussians at each output [105] that *explicitly* parametrise the proposal distribution. In this toy example, both approaches seem to learn proposals that are close to the true posterior.

6.1.5.2 A benchmark nonlinear state-space model

We now consider a more complex temporal model given by the following dynamics

$$\begin{aligned} &p(x_n|x_{n-1}) = \mathcal{N}(x_n; f(x_{n-1}), \sigma_v^2), p(x_1) = \mathcal{N}(x_1; 0, 5) \\ &p(y_n|x_n) = \mathcal{N}(y_n; g(x_n), \sigma_w^2) \\ &f(x_{n-1}) = x_{n-1}/2 + 25x_{n-1}/(1+x_{n-1}^2), g(x_n) = x_n^2/20 \end{aligned} \tag{6.14}$$

for fixed $\theta = (\sigma_v, \sigma_w) = (\sqrt{10}, 1)$. The nonlinear model, depicted graphically as a state space hidden Markov model in Figure 6.3 (b), is often used to assess the performance of SMC methods [112, 31]. The model satisfies the *Markov property* and is *time-invariant*, that is, $p(x_n|x_{n-1})$ and $p(y_n|x_n)$ are independent of n, hence we only train a single proposal distribution $q_\phi(x_n|x_{n-1}, y_n)$ according to (6.4) and we use it for sampling x_n for all $n = 1, ..., N$. The posterior density $p_\theta(x_{1:N}|y_{1:N})$ is highly multimodal due uncertainty about the sign of the state x_n which is only observed through its square.

We use a simple two-layer network with 300 hidden units in each layer for the proposal and a five-layer network with 300 hidden units in each layer for the discriminator. We adversarially train them using a dataset of 100 points sampled according to the generative process in (6.14). We compare the performance of our approach (AVB-PF) against bootstrap particle filters (PF) as well as particle filters with a proposal represented by a neural network of the same dimensions as our proposal, but with density *explicitly* parametrised by a mixture of three Gaussians at each output (NN-PF) [105]. In contrast to NN-PF, we include the noise ε as additional input to the proposal model instead of adding it at the very end, thereby

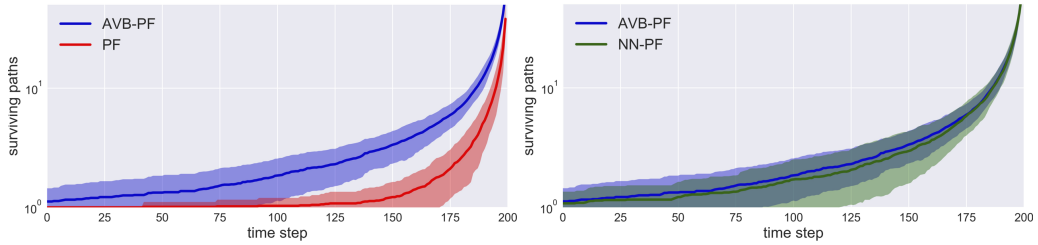


Figure 6.5: Adversarially learnt proposals reduce particle degeneracy in the nonlinear state-space model. Here we show the number of unique particles which survive over the course of 200 time steps, using 100 particles. Plots show mean and standard deviation over 100 sampled sequences.

allowing the proposal network to learn complex probability distributions. Results show improved performance in terms of sample quality and marginal log likelihood (see Figures 6.5 and 6.6).

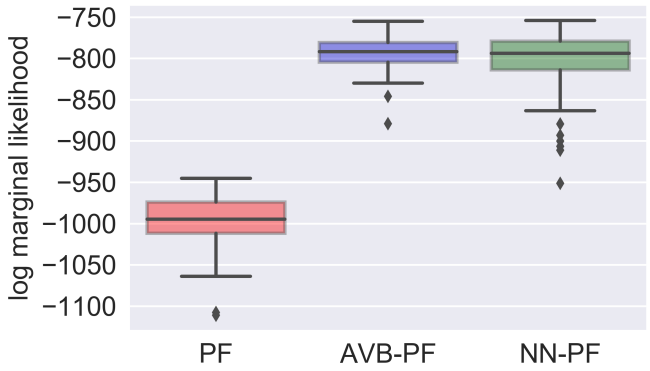


Figure 6.6: Box-plots for log-marginal likelihood estimates over 100 sampled sequences of length $N = 200$, using 100 particles. Our inference method explains data better, both in terms of median log-likelihood and its variability across different data samples.

6.1.5.3 An indoor navigation challenge

Finally, consider a real case study of an indoor navigation challenge hosted at Kaggle[1]. A person moves between five locations in a room and at each location records distances to three bluetooth beacons A, B, C using a bluetooth sensor in their smartphone device (see Figure 6.7). Our goal is to infer a sequence of positions visited by a person, $x_{1:N}$, from a sequence of collected measurements $y_{1:N}$.

We represent the problem as a Hidden Markov Model (introduced in Sec-

tion 3.3.1.1) with discrete latent variables x_n encoding the position of a person at time n and continuous observed variables $y_n = (dist_A, dist_B, dist_C)$ encoding distances to the three beacons. We use a 1-of-K coding scheme for the latent variables, which turns x_n into K-dimensional binary variables and assume a Gaussian noise model for distances measured between each position and each bluetooth beacon. This model, whose graphical structure is shown in Figure 6.3 (b), can be represented as

$$\begin{aligned} p(x_n|x_{n-1,A}) &= \prod_{k=1}^{K}\prod_{j=1}^{K} A_{jk}^{x_{n-1,j},x_{n,k}} \\ p(y_n|x_n) &= \prod_{d=1}^{D}\prod_{k=1}^{K} \mathcal{N}(y_n^d|\mu_{dk},\Sigma_{dk})^{x_k} \end{aligned} \tag{6.15}$$

where A denotes the transition matrix such that $A_{jk} = p(x_{n,k} = 1|x_{n-1,j} = 1)$ is the probability of a person transitioning from position j to position k, μ is the matrix of mean distances between beacon d and position k, and Σ is the covariance matrix that models the spread of distances for each position and beacon pair. We infer model parameters A, μ and Σ from actual observations with true positions labelled[1].

We train our method to be able to predict actual positions $x_{1:n}$ given distances to beacons $y_{1:N}$. As in Section 6.1.5.2, we use a two-layer network for the proposal and a five-layer network for the discriminator. We use fifty data pairs $\{x_{n-1}^{(l)}, y_n^{(l)}\}$ for training, ten for each of the five possible positions x_n. We test our method using the remaining dataset $\{x_{1:N}, y_{1:N}\}$ of length $N = 236$. We repeat the test 100 times to get the performance metrics summarised in Table 6.1.

Our method shows high performance despite training on a very small data sample. This is contrary to NN-PF, which completely fails in this real case study. Interestingly, when we repeat the same experiment but on data synthetically generated according to (6.15) instead of real data, NN-PF performs at least as well as our method (see Table 6.2). The difference in performance stems from the fact that real data does not ideally fit our modelling assumptions, especially our Gaussian noise

[1]Kaggle indoor positioning challenge: `https://www.kaggle.com/liwste/indoor-positioning`

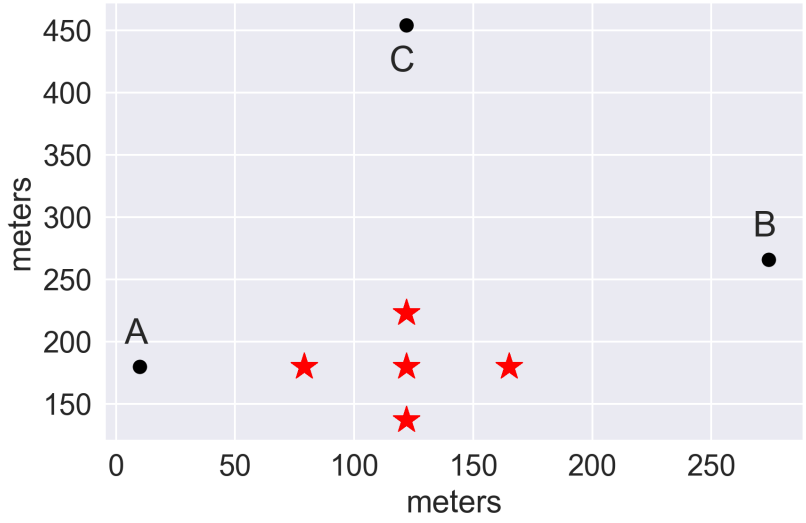


Figure 6.7: Experimental outline for the indoor navigation challenge. A person moves between five positions (marked in red) and measures distances to bluetooth beacons A, B and C.

Table 6.1: ESS, LML and percentage match in the indoor navigation challenge. Table shows mean and standard deviation over 100 runs on a test trajectory of length $N = 236$ using 50 particles. Percentage match is calculated as a fraction of positions that are correctly identified.

	ESS (ITER)		LML		% MATCH	
	MEAN	STD	MEAN	STD	MEAN	STD
PF	47.91	6.70	443.55	216.90	95.83	8.20
AVB-PF	28.75	5.81	**509.47**	4.63	**99.58**	0.65
NN-PF	46.75	6.27	-841.50	88.82	51.30	5.14

model. Our implicit method shows more robustness against such inconsistencies. Notice also that effective sample size (ESS) is a poor indicator of actual performance in Table 6.1. It only measures relative quality of samples, therefore, it may lead to inaccurately high scores when all samples are similar *and* of poor quality.

6.1.6 Discussion

We present a new method for training proposal models for sequential Monte Carlo based on adversarial training. This allows us to make the proposal model much more flexible, effectively allowing it to represent any family of conditional distributions over the latent variables.

Table 6.2: ESS, LML and percentage match using *synthetic* indoor navigation data. Table shows mean and standard deviation over 100 runs on a test trajectory of length $N = 236$, using 50 particles.

	ESS (ITER)		LML		% MATCH	
	MEAN	STD	MEAN	STD	MEAN	STD
PF	44.71	6.51	-2600.68	34.81	99.69	0.41
AVB-PF	49.69	2.48	-2545.48	0.35	**99.99**	0.04
NN-PF	49.84	2.46	**-2544.73**	0.33	99.98	0.09

We view this work primarily as a way to automate the design of high-quality proposal models, however, the neural network itself can be seen as the desired artifact, in the sense of variational autoencoders [113]; direct sampling from the proposal may provide a satisfactory inference approximation even omitting the importance weighing or SMC steps.

We believe that further progress can be made by validating the method on inference problems with larger volumes of data and by improving the stability of GAN training through regularization [114] or by replacing the discriminator with statistical two-sample testing [115] (see section below). Future work should also address the simplistic assumption that all proposal samples come from a uniform distribution. More appropriate ways of density estimation could use annealed importance sampling [120] or a discriminator network in GANs (see Chapter 7 for more details).

6.2 Moment matching variational Bayes

6.2.1 Introduction

The promise of deep generative models is to generate compelling data samples from rich, multimodal probability distributions over the kinds of data describing the world around us, such as natural images, written text or audio waveforms. Generative adversarial networks (GANs) [102] provide a an algorithmic framework for training deep generative models with several appealing properties: they do not require a data likelihood function to be specified, only a data generating procedure;

they provide samples that are sharp and compelling; they do not rely on Markov chains or approximate inference networks for neither training nor generation of samples.

A deep generative model in GANs is trained by introducing an extra discriminative network, which tries to distinguish between generated samples and data samples. The generative network is then trained to counteract this in order to make the samples indistinguishable to the discriminator. The gradient of the objective can be backpropagated through the generative network. However, because of the difficult minimax nature of the objective, backpropagation can easily get stuck at a local optima. To address this shortcoming, an alternative framework for training deep generative models has been proposed based on a technique from statistical hypothesis testing known as maximum mean discrepancy (MMD). The framework, called generative moment matching networks (GMMNs) [116], can be interpreted as matching all moments of distributions between data samples and samples from the model. MMD leads to a simpler loss function that can be easily trained by backpropagation.

In this research, we introduce generative model matching networks for improved variational inference. We use deep generative models to represent highly expressive inference models, which we train by rephrasing variational inference as MMD distance minimisation. Our research leads to significant improvements in the accuracy of variational inference, as tested within Variational Autoencoders (VAEs). It also benefits from improved training stability in comparison to alternative approaches to variational inference with deep generative models based on GANs (see Section 6.1).

6.2.2 Methodology

6.2.2.1 Variational inference

A directed graphical model with latent variables x and observed variables y specifies a factorisation of the joint distribution $p(x,y)$ in terms of a prior $p(x)$ over the latent variables and a parametric generative model $p_\theta(y|x)$. Our goal is to learn an approximation $q_\phi(x|y)$ to the intractable true posterior $p(x|y)$ as well as to estimate

the marginal likelihood or model evidence

$$E_{p_D(y)} \log p_\theta(y) \tag{6.16}$$

where p_D is the data distribution. Although this problem is usually intractable, we can derive an approximation by rewriting the marginal likelihood as:

$$\begin{aligned} \log p_\theta(y) &= \log \int_x p_\theta(y,x) \\ &= \log \int_x p_\theta(y,x) \frac{q_\phi(x|y)}{q_\phi(x|y)} \\ &= \log \left(E_{q_\phi(x|y)} \left[\frac{p_\theta(x,y)}{q_\phi(x|y)} \right] \right) \end{aligned} \tag{6.17}$$

$$\geq E_{q_\phi(x|y)} \left[\log \frac{p_\theta(x,y)}{q_\phi(x|y)} \right] \tag{6.18}$$

The right hand side of (6.18) is called the variational lower bound or evidence lower bound (ELBO). If there is ϕ such that $q_\phi(x|y) = p(x|y)$, we would have

$$\log p_\theta(y) = E_{q_\phi(x|y)} \left[\log \frac{p_\theta(x,y)}{q_\phi(x|y)} \right] \tag{6.19}$$

However, this is generally not the case, so we are left with an inequality in (6.19). The inequality enables us to rephrase the intractable problem in (6.16) into

$$\max_\theta \max_\phi E_{p_D(y)} E_{q_\phi(x|y)} \left[\log \frac{p(x,y)}{q_\phi(x|y)} \right] \tag{6.20}$$

or equivalently into

$$\max_\theta \max_\phi E_{p_D(y)} E_{q_\phi(x|y)} (\log p(x) - \log q_\phi(x|y) + \log p_\theta(y|x)). \tag{6.21}$$

The optimisation in (6.21) allows us to learn parameters ϕ^* such that $q_{\phi^*}(x|y)$ is close to the true posterior $p(x|y)$. Naturally, the quality of the learning depends on the expressiveness of the approximate inference model $q_\phi(x|y)$.

For example, in Variational Autoencoders (VAEs) [113, 117], which assume a

simple generative model where each observation y_n has an associated latent variable x_n (see Figure 6.8), $q_\phi(x|y)$ is usually specified as a Gaussian distribution with mean and variance parametrized by neural networks with y as input [113, 117]. This specification might be too restrictive in cases where the posterior distribution is assumed to be highly multimodal, such as in the case of natural images, where VAEs have often resulted in blurry images [118].

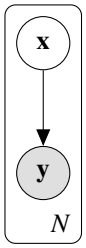


Figure 6.8: A directed graphical model of variational autoencoders.

6.2.2.2 Implicit inference models

In this work, we propose the use of deep generative models to represent highly flexible inference models $q_\phi(x|y)$. The inference models then become *implicit* probabilistic models, that is, models that we can sample from but whose marginal likelihood we cannot evaluate.

Implicit inference models use an input noise variable $g(\varepsilon)$ and a deep network architecture x_ϕ parametrised by ϕ to flexibly approximate any posterior distribution over latent variable x given observation y:

$$x = x_\phi(y, \varepsilon'), \qquad \varepsilon' \sim g(\varepsilon) \tag{6.22}$$

6.2.2.3 Learning objective

Since implicit inference models do not have an explicit parametric form, we cannot use the reparametrisation trick [113, 119] and stochastic gradient descent to find parameters ϕ that optimise (6.21). We could circumvent the problem by implicitly representing the term in (6.21)

$$\log p(x) - \log q_\phi(x|y) \tag{6.23}$$

as the optimal value of a logistic regressor, a discriminator network, that we would introduce to the problem. This would, however, lead to a difficult two-player objective, which is prone to training instability, as experienced in our previous work in Section 6.1.

Instead, we propose to replace the problem of *density ratio* estimation in (6.23) with *density distance* estimation proposed by [116] as moment matching. This is an alternative training approach based on the principle of density comparison [107]. The approach leads to a much simpler learning objective, minimizing the maximum mean discrepancy (MMD), which can be steadily optimised using gradient descent. Sriperumbudur et al. [120] shows that density estimation using moment matching can be directly related to class-probability estimation by representing MMD as an integral probability metric.

The MMD metric is defined as follows. Suppose we are given two sets of samples $X = \{x_i\}_{i=1}^N$ and $Y = \{y_j\}_{j=1}^M$ and are asked whether the generating distributions $P_X = P_Y$. Maximum mean discrepancy answers this question by comparing statistics between the two datasets. Formally, the following MMD measure computes the mean squared difference of the statistics of the two datasets:

$$MMD^2(X,Y) = \frac{1}{N^2}\sum_{i=1}^{N}\sum_{i'=1}^{N} k(x_i,x_{i'}) - \frac{2}{NM}\sum_{i=1}^{N}\sum_{j=1}^{M} k(x_i,y_i) + \frac{1}{M^2}\sum_{j=1}^{M}\sum_{j'=1}^{M} k(y_j,y_{j'}) \tag{6.24}$$

where $k(x,x')$ is a kernel function which implicitly lifts the sample vectors into an infinite dimensional feature space. When this feature space corresponds to a universal reproducing kernel Hilbert space, it is shown that $MMD = 0$ if and only if $P_X = P_Y$. For universal kernels like the Gaussian kernel, defined as

$$k(x,x') = \exp(-\frac{1}{2\sigma}|x-x'|^2), \tag{6.25}$$

where σ is the bandwidth parameter, minimizing MMD is equivalent to minimizing a distance between *all* moments of the two distributions.

We replace the density ratio estimation in (6.23) with the density distance estimation in (6.24) and use (6.22) to rewrite the optimisation objective in (6.21) as

Algorithm 4 Moment-matching variational Bayes

$i \leftarrow 0$;
while not converged **do**
  Sample $\boldsymbol{y} = \{y^{(1)}, ..., y^{(m)}\}$ from data distrib. $p_D(y)$;
  Sample $\boldsymbol{x} = \{x^{(1)}, ..., x^{(m)}\}$ from prior $p(x)$;
  Sample $\boldsymbol{\varepsilon} = \{\varepsilon^{(1)}, ..., \varepsilon^{(m)}\}$ from $\mathcal{N}(0,1)$;
  Sample $\boldsymbol{x_\phi} = \{x_\phi(y^{(1)}, \varepsilon^{(1)}), ..., x_\phi(y^{(m)}, \varepsilon^{(m)})\}$ according to eq. 6.22;
  Compute θ-gradient (eq. 6.26):
  $g_\theta \leftarrow \frac{1}{m}\sum_{l=1}^{m} \nabla_\theta \left[\log p_\theta(y^{(l)}|x_\phi(y^{(l)}, \varepsilon^{(l)}))\right]$;
  Compute ϕ-gradient (eq. 6.26):
  $g_\phi \leftarrow \nabla_\phi \left[-MMD^2(\boldsymbol{x_\phi}, \boldsymbol{x}) + \frac{1}{m}\sum_{l=1}^{m} \log p_\theta(y^{(l)}|x_\phi(y^{(l)}, \varepsilon^{(l)}))\right]$;
  Perform SGD-updates for θ and ϕ: $\theta \leftarrow \theta + h_i g_\theta$ $\phi \leftarrow \phi + h_i g_\phi$
  $i \leftarrow i + 1$;
end while

$$\max_\theta \max_\phi E_{p_D(y)} E_\varepsilon [-MMD^2(x_\phi(y, \varepsilon), p(x)) + \log p_\theta(y|x)] \tag{6.26}$$

which is optimised according to Algorithm 4. It can be easily shown that if there are parameters ϕ^*, such that $q_\phi(x|y) = p(x)$, then the density ratio estimation in (6.23) and the density distance estimation in (6.24) are both equal to zero. Hence, the optimisation objectives in (6.21) and (6.26) are equivalent. Further work could investigate whether we could explicitly derive the proposed objective in (6.26) from the problem of marginal likelihood estimation in (6.16).

6.2.3 Experiments

Throughout this section we evaluate the effect of using moment matching-based posterior approximations as inference model in variational autoencoders (VAEs).

Kernel function: Our approach requires evaluating a kernel function $k(x, x')$ in (6.24). We use a Gaussian kernel defined in (6.25). The kernel has a bandwidth parameter σ which plays a crucial role in determining the statistical efficiency of moment-matching based learning. Finding an optimal σ is an open problem, so as a simple approximation, we use a mixture of K kernels spanning multiple ranges. That is, we choose the kernel to be:

$$k(x, x') = \sum_{p=1}^{K} k_{\sigma_p}(x, x') \tag{6.27}$$

Figure 6.9: Training images in the synthetic dataset.

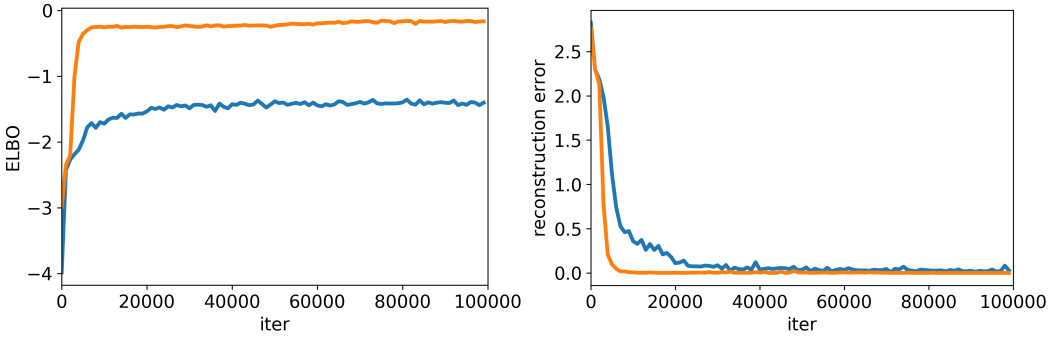
where k_{σ_p} is a Gaussian kernel with bandwidth parameter σ_p. In our experiments, we use a mixture of six kernels with $\sigma_p \in \{2, 5, 10, 20, 40, 80\}$. Further work could investigate kernel mixtures with other bandwidth values.

Synthetic example: We illustrate the application of our method to learning a generative model on a toy example proposed by Mescheder et al. [109]. We are given a simple dataset containing only 4 data points from the space of $2x2$ binary images in Figure 6.9 and a two-dimensional latent space. Our goal is to learn the division of the latent space corresponding to the 4 possible observed images.

We introduce the encoder and the decoder network, each parametrized by 2-layer fully connected neural networks with 512 hidden units each. The encoder, which is our moment matching-based inference model, takes as input a data point y and a Gaussian random noise ε and produces a latent value z. The decoder takes as input a latent value z and outputs the parameters of four Bernoulli distributions, one for each pixel of the output image.

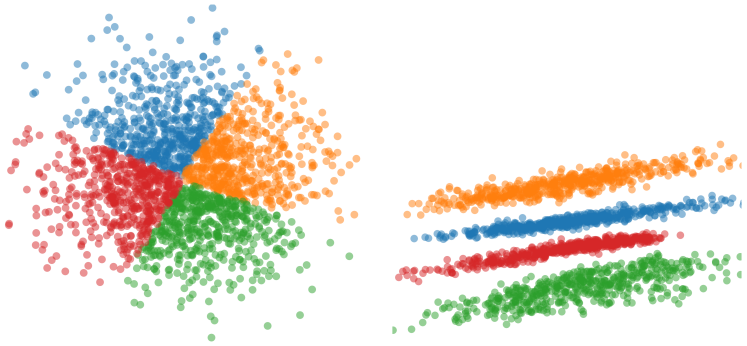
We compare our method, moment matching variational Bayes or MMVB in short, against state-of-the-art variational autoencoder proposed by Mescheder et al. (AVB) [109] which uses exactly the same encoder and decoder network architecture, but trains the encoder adversarially using a discriminator network instead of using moment matching.

Our method trains faster and results in a better generative model as shown in Figure 6.10. In particular, we see that the reconstruction error given by the mean cross-entropy between an input x and its reconstruction using the encoder and decoder networks is slightly lower when using moment-matching based training for the encoder network. We also observe that the estimated variational lower bound is much lower in our method. This result, however, should be treated with caution as we should still compare it against the true log-likelihood to make sure the



(a) ELBO **(b)** reconstruction error

Figure 6.10: Our method (orange) in comparison to AVB (blue) leads to much faster training and improved ELBO and reconstruction error on the synthetic dataset.



(a) AVB **(b)** MMVB

Figure 6.11: Distribution of latent variable x for AVB and MMVB trained on the synthetic dataset.

approximation is close to the true value.

We visualise the learnt division of the latent space in Figure 6.11, where each colour corresponds to one state in the x-space. Interestingly, our method learns a lower dimensional representation of the latent space, corresponding to one-dimension along the y-axis. This shows the ability of our method to find underlying structure in the observed images. The resulting latent space, however, deviates from the assumption of two-dimensional Gaussian prior $p(x)$, which makes it difficult to efficiently sample latent codes for new images.

MNIST: In addition, we validate our method on a more advanced problem of digit generation. We represent the encoder and the decoder as 5-layer deep convolutional networks based on the DC-GAN architecture [121]. We train them on the binarized MNIST dataset [122]. Figure 6.12 shows random samples from the trained de-

	RECONSTRUCTION ERROR	ELBO	
MMVB	**81.9**	**-83.0**	
VAE	112.6	-128.1	
AVB	84.3	-95.7	MESCHEDER ET AL. [109]
AVB + AC	91.9	-109.7	MESCHEDER ET AL. [109]
VAE + IAF	223.2	-223.4	KINGMA ET AL. [124]
AUXILIARY VAE	93.9	-110.0	MAALØE ET AL. [125]

Table 6.3: Comparison of our method (MMVB) and other methods improving on VAEs. We see that our method achieves state-of-the-art ELBO and reconstruction error for a 8-dimensional latent space on binarized MNIST.

(a) random samples **(b)** interpolation

Figure 6.12: Our model trained on MNIST produces perceptually good (a) random samples and (b) interpolation when the latent input code x is gradually moved between latent codes of two images in the training data.

coder network. We compare our method (MMVB) against variational autoencoder with Gaussian inference model as well as improved variational autoencoders with more expressive inference models. Results summarized in Table 6.3 shows state-of-the-art performance of our method both in terms of ELBO, a lower bound on the marginal log likelihood, and the reconstruction error. Further validation could also look at log-likelihoods achieved by decoder networks of the different methods, approximated using Annealed Importance Sampling (AIS) [123], which would validate whether strong inference models are actually necessary to obtain a good generative model.

6.2.4 Discussion

We present a new training procedure for variational autoencoders based on moment matching. The procedure enables training highly flexible inference models as deep generative models. In contrast to an alternative training regime based on adversarial training (AVB), our method leads to improved training stability and a better generative model.

Our method learns low-dimensional latent representations, which correctly capture the complexity of the observed data. Further work is underway to understand how this property could be exploited, e.g. for efficient data sampling.

Future work could also apply our method to train highly flexible proposal models for sequential Monte Carlo. This would present a more stable alternative to adversarial training in our work on adversarial Sequential Monte Carlo (Section 6.1).

6.3 Conclusions

Driven by questions on urban dynamics, the chapter proposes novel approaches to approximate inference in probabilistic graphical models that borrow ideas from deep generative models (GANs, GMMNs) to train highly flexible inference models.

In Section 6.1, a deep generative model represents a proposal model for a sequential importance sampler, such as particle filter. The model is adversarially trained to closely approximate the desired posterior, hence establishing a principled connection between GANs and SMC.

In Section 6.2, a deep generative model becomes an inference model (encoder) within a variational autoencoder. This time, it is trained using a statistical learning method of moment matching (GMMNs), which shows higher training stability than the two-player training of GANs.

The research projects are examples of Bayesian deep learning that utilize deep generative models within Bayesian inference as flexible approximations to highly non-linear posterior distributions. The generative models learn near-optimal inference models directly from data, hence they remove the need to manually design appropriate inference distributions. The models can approximate arbitrarily com-

plex distributions, hence they also overcome the expressiveness limit of parametric distributions, such as mixtures of Gaussians.

The idea of using deep generative networks as inference models is not without problems, however. The first problem stems from the fact that it is impossible to evaluate the likelihood of samples from deep generative models. The likelihood estimation is needed to assign weights to samples within an importance sampler, such as particle filter in Section 6.1. More broadly, it is necessary to evaluate the performance of a generative model. So far in Section 6.1, we have made a crude assumption that the samples are equally likely under the generative model. Future work will lift this assumption and investigate more appropriate ways of density estimation using annealed importance sampling [126] or a discriminator network in GANs.

The second problem relates to the difficulty of training deep generative models. Recent research has introduced techniques that encourage convergence of adversarial training in GANs [127], as well replace the adversarial objective in GANs with a simpler training objective based on feature matching. Our work in progress in Section 6.2 shows promising results of training an inference model based on feature matching. This research will be continued.

Chapter 7

General Conclusions and Outlook

The ever growing urbanisation presents us with a real need to understand cities now and in the nearest future. We need to be able to *reason* about cities: to take the available data and reach conclusions, both about what might be true about city dynamics now and how to act. Cities are becoming ”smart”, recording their daily pulses in large databases and presenting us with an unprecedented opportunity to reason about them directly from data.

This thesis proposes a general framework for knowledge discovery about cities from urban tracking data. The framework is based on the concept of *declarative representation*. In this approach, we create a *model* that encodes our knowledge about how urban dynamics happen. We then apply a reasoning algorithm that takes available data to answer questions based on the model. The declarative representation is one of the most promising approaches towards enabling a computer program to reason. The intuition behind this approach follows a famous quote from Richard Feynman: ”What I cannot create, I cannot understand”. The model declares how urban dynamics are *created* and any unknown quantities in the model are learnt from observations using the reasoning algorithm. For example, a model of how drivers navigate in cities might represent our knowledge about how navigation is influenced by regions in cities. A reasoning algorithm can take this model, as well as observations of drivers' routes, and learn regionalisation of cities as perceived by drivers. Our framework is fully probabilistic, which enables a systematic approach to reasoning under uncertainty. Uncertainty appears to be an inescapable aspect

of most urban applications. It might arise through noise in urban measurements, limited size of data sets, or uncertainty about the best model to explain data.

The proposed framework is exemplified by answering two questions on urban dynamics:

1. **Where you are**: can we infer true locations from noisy tracking data?

2. **How you navigate**: can we extract urban regions, as perceived by drivers, from their tracking data?

Each question is approached by defining a model that encodes our knowledge about the underlying dynamics. The model is then coupled with an inference algorithm to answer the questions. For example, to infer true locations from tracking data, we use a state space hidden Markov model to encode the relationship between true locations and the corresponding noisy observations. We then propose novel inference algorithms to learn the true locations from the data.

7.1 Summary of contributions

The thesis advances our ability to answer questions about urban dynamics and our more general toolkit of Bayesian inference in probabilistic graphical models.

Firstly, to address the question of *where you are*, the thesis proposes three algorithms for map-matching, i.e. inferring locations on the road network given noisy tracking data. The algorithms are fully probabilistic, hence they are capable of expressing map-matching confidence. They show various levels of map-matching flexibility at the expense of computational efficiency. On one extreme, PST-Matching algorithm is proposed that limits possible true locations to a few locations on the road network but offers high computational efficiency. On the other extreme, a bootstrap particle filter algorithm is shown that considers any positions on the road network as possible candidates, but might be prohibitively slow, especially when observations are sparse (due to a large number of particles needed). Finally, look-ahead particle filter is outlined that offers *both* high efficiency and flexibility in the special case of sparse tracking data.

Secondly, to address the question of *how you navigate*, a novel approach to regional delineation using topic modelling is proposed. Topic modelling discovers fine-grained route preferences through different neighbourhoods by encoding route choice theory as a generative model. The discovered routes shed a light on the partition of the urban space as perceived by drivers. They also enable accurate predictions of routes taken between two locations.

Finally, the thesis proposes novel algorithms for Bayesian inference in probabilistic graphical models which extend beyond the motivating questions on urban dynamics. The algorithms borrow ideas from deep generative models (GANs, GMMNs) to enable inference with highly non-linear and multimodal posterior distributions. The algorithms, although only validated on standard benchmark datasets in the thesis, could be applied in the future to questions on urban dynamics and beyond.

7.2 Vision for the future

Our long-term research goal is to harness model-based machine learning, fulfilled as Bayesian inference in graphical models, to understand, predict, and ultimately enhance cities. We would like to create explanatory models of urban dynamics and robust inference algorithms which together enable learning about urban dynamics from data. We would also like to make model-based machine learning as approachable as possible to increase its uptake by researchers in urban data science.

The long-term goal requires further research on two fronts. On one front, we would like to develop more model-based answers to questions about urban dynamics. We could continue our work by adding additional data sources and improving the granularity of the explanatory models. As a short term goal, the question of *where you are* could be revisited since with the release of Android 7, phone users are given access not only to the phone's position estimate, but also the signal-to-noise ratio (SNR) for each GNSS satellite (global navigation satellite systems) in view. If we could feed this "signal strength" information together with 3D maps into an inference algorithm, we could further refine our location estimates.

On another front, we would like to make model-based learning easier to use for researchers from applied domains, such as urban science. Model-based machine learning is based on the declarative representation which clearly separates knowledge and reasoning. It has been designed with the premise that newcomers to the field of machine learning would not have to learn about a whole suite of algorithm, but instead focus their attention on understanding a single modelling environment [128]. Once they encoded their knowledge as a model, an 'off-the-shelf' inference algorithm would then take care of learning about the modelled system from available data. This vision is still in its infancy and requires further research into automated inference algorithms. We would like to contribute to this stream of research by further investigating deep generative models as highly expressive inference models. As a short term goal, we would like to continue our work on approximate inference using adversarial sequential Monte Carlo. The algorithm requires a more appropriate approach to sample weighing that could use annealed importance sampling or a decoder network.

7.2.1 Short term goals

7.2.1.1 Improved localisation using satellite signal strength

Future work could explore the use of satellite signal strength as a valuable source of location information. This is a timely research idea since with the release of Android 7, Android phones now provide not just the phone's position estimate, but also the signal-to-noise ratio (SNR) for each GNSS satellite in view. If we combine this "signal strength" information together with 3D maps, we could obtain very detailed location information. For example, we could distinguish between different sites of the street if we know signal reflections caused by buildings on both sites of the street.

The intuition behind using using signal loss as an information source is as follows. If the SNR for a satellite is low, then the line-of-sight is probably blocked or shadowed; if the SNR is high, then the line-of-sight is probably clear. Notice that the qualifier "probably" could be intuitively expressed using probability distributions in model-based machine learning.

We could encode the intuition on signal strengths within a probabilistic graphical model. For each satellite, we could check whether the ray from the satellite to any possible location is blocked using our 3D map. We could then use the line-of-sight and shadowed conditions to determine the most likely signal strength over possible locations for each satellite. Then the overall likelihood of a location, based on the satellite signal strengths, would be the product of the likelihoods corresponding to the different satellites. This approach would create a probability distribution over possible receiver locations, based on satellite signal strengths alone.

We could further refine the location estimate by fusing this information with position estimate, as well as past measurements, inside a sequential Monte Carlo algorithm, such as particle filter. The particle filter should estimate true location better than our previous work in Section 4.5.2 which does not consider signal strength and 3D map information.

7.2.1.2 Improved importance weights for adversarial sequential Monte Carlo

Our research in Section 6.1 proposes the use of deep generative models as highly flexible proposal models for sequential Monte Carlo. The proposals can approximate highly non-linear posterior distributions, but, on the downside, they do not lend themselves to density evaluation which leaves the open question of how to weigh their samples. So far in Section 6.1 we have made the crude assumption that all samples from the implicit proposals have uniform density (see Section 6.1.3.3). This is a simplistic assumption that should be addressed in future work.

The ability to evaluate the likelihood of samples from an implicit generative model is needed to assign importance weights in adversarial sequential Monte Carlo, but also more generally to measure the quality of performance of implicit models. The most widely used estimator of log-likelihood for GANs is the Kernel Density Estimator (KDE) [129] which assumes a Gaussian observation model p_σ with a fixed variance hyperparameter σ^2. The density estimate of the proposal distribution $q(x|y)$ of the latent variables x given the observed data variables y would then become

$$q(x|y) = \frac{1}{K}\sum_{k=1}^{K} p_\sigma(x|y, \varepsilon^{(k)}) \tag{7.1}$$

where $\{\varepsilon^{(k)}\}_{k=1}^{K}$ are samples from the input noise $g(\varepsilon)$. Unfortunately, KDE is notoriously inaccurate for estimating likelihood in high dimensions, because it is hard to cover a high-dimensional manifold with spherical Gaussians [130].

Instead, we would like to investigate the use of the discriminator network in adversarial SMC for importance weighing. Recall that the importance weights in SMC (3.20) are defined recursively as

$$w(x_{1:N}) = w_1(x_1)\prod_{n=2}^{N} w_n(x_{1:n}) \tag{7.2}$$

where

$$\begin{aligned} w_1(x_1) &= \frac{p(x_1)p(y_1|x_1)}{q_1(x_1|y_1)} \\ w_n(x_{1:n}) &= \frac{p(x_n|x_{1:n-1})p(y_n|x_{1:n}, y_{1:n-1})}{q_n(x_n|x_{1:n-1}, y_{1:n})} \end{aligned} \tag{7.3}$$

Since the intermediate proposals q_n are represented as implicit models in adversarial SMC, we cannot evaluate their density in (7.3). So far, we have addressed this problem by assuming a uniform density across their samples, effectively removing the term q_n from the importance weight calculation in (7.3).

In future work, we want to lift this assumption and approximate $q_n(x_n|x_{1:n-1}, y_{1:n})$ for each proposal sample x_n using the discriminator network. Recall that the optimal discriminator T^* in adversarial SMC is given by

$$T^*(x_{1:n}, y_{1:n}) = \log \frac{q_n(x_n|x_{1:n-1}, y_{1:n})}{p(x_n|x_{1:n-1})}. \tag{7.4}$$

If we assume that after training the discriminator approaches the optimal discriminator T^*, we could rearrange (7.4) to approximate the intermediate proposal densities as follows

$$q_n(x_n|x_{1:n-1}, y_{1:n}) = p(x_n|x_{1:n-1})\exp(T^*(x_{1:n}, y_{1:n})). \tag{7.5}$$

Then the importance weights in (7.3) would become

$$\begin{aligned} w_1(x_1) &= \frac{p(y_1|x_1)}{\exp(T^*(x_1, y_1))} \\ w_n(x_{1:n}) &= \frac{p(y_n|x_{1:n}, y_{1:n-1})}{\exp(T^*(x_{1:n}, y_{1:n}))}. \end{aligned} \tag{7.6}$$

The proposed approach uses the discriminator network in GANs to properly evaluate importance weights in adversarial sequential Monte Carlo. More broadly, it provides a tool for an accurate evaluation of sample quality in GANs, which could benefit a range of applications.

Chapter 8

Publications

Parts of the thesis have been published in a number of journal and conference proceeding papers. Below is a list of the publications grouped by the chapter that they correspond to.

Chapter 4: Where you are

1. Kira Kempinska, Toby Davies, John Shawe-Taylor, and Paul Longley. Probabilistic map-matching for low-frequency gps trajectories. In *Proceedings of GIS Ostrava conference*, pages 209–221. Springer, 2017. This paper features the probabilistic ST-Matching algorithm.

2. Kira Kempinska, Toby O. Davies, and John Shawe-Taylor. Probabilistic map-matching using particle filters. In *Proceedings of the 24th GIS Research UK conference*, 2016. This paper described the basic implementation of particle filters for probabilistic map-matching.

3. Kira Kempinska and John Shawe-Taylor. Improved particle filters for vehicle localisation. In *NIPS 2016 Advances in Approximate Inference workshop*, 2016. This paper introduced the look-ahead particle filters algorithm.

Chapter 5: How you navigate

1. Kira Kempinska, Paul Longley, and John Shawe-Taylor. Interactional regions in cities: making sense of flows across networked systems. *International*

Journal of Geographical Information Science, 32(7):1348–1367, 2018. This journal paper describes our methodology for extracting interactions regions from vehicle tracking data.

2. Kira Kowalska, John Shawe-Taylor, and Paul Longley. Data-driven modelling of police route choice. In *Proceedings of the 23rd GIS Research UK conference*, 2015. The early paper explores the use of interactional regions for route prediction.

3. T Cheng, Kate Bowers, Paul Longley, John Shawe-Taylor, Trevor Adams, Toby Davies, Gabriel Rosser, Sarah Wise, Chris Gale, Monsuru Adepeju, Jianan Shen, Huanfa Chen, Dawn Williams, Kira Kempinska, and Artemis Skarlatidou. *CPC: Crime, Policing and Citizenship - Intelligent policing and big data.* 05 2016. This book includes our work on police route choice modelling.

Chapter 6: Methodological extensions

1. Kira Kempinska and John Shawe-Taylor. Adversarial sequential monte carlo. In *NIPS 2017 Advances in Approximate Inference workshop*, 2017. Our most recent work that won the contributed talk and the Spotify award.

Bibliography

[1] Michael Batty. *The New Science of Cities*. MIT Press, 2013.

[2] Yin Lou, Chengyang Zhang, Yu Zheng, Xing Xie, Wei Wang, and Yan Huang. Map-matching for low-sampling-rate GPS trajectories. In *Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems - GIS '09*, page 352, New York, New York, USA, November 2009. ACM Press.

[3] Jane Jacobs. The death and life of great american. *Cities*, pages 321–25, 1961.

[4] John Winn, Christopher Bishop, and Thomas Diethe. *Model based machine learning*. 2018.

[5] Dawn Woodard, Galina Nogin, Paul Koch, David Racz, Moises Goldszmidt, and Eric Horvitz. Predicting travel time reliability using mobile phone gps data. *Transportation Research Part C: Emerging Technologies*, 75:30–44, 2017.

[6] PC Gregory. A bayesian analysis of extrasolar planet data for hd 73526. *The Astrophysical Journal*, 631(2):1198, 2005.

[7] Zoubin Ghahramani. Probabilistic machine learning and artificial intelligence. *Nature*, 521(7553):452–459, May 2015.

[8] E. T. Jaynes. *Probability Theory: The Logic of Science*. 2003.

[9] Richard T. Cox. *Algebra of Probable Inference*. Johns Hopkins University Press, 1961.

[10] Kevin S Van Horn. Constructing a logic of plausible inference: a guide to Cox's theorem. *International Journal of Approximate Reasoning*, 34(1):3–24, September 2003.

[11] B. De Finetti. La prévision: ses lois logiques, ses sources subjectives. *Annales de I'institut Henri Poincaré [in French]*, 7:1–68, 1937.

[12] C.E. Rasmussen and Zoubin Ghahramani. Occam's Razor. *Neural Information Processing Systems*, 13:294–300, 2001.

[13] W. Jefferys and J. O. Berger. Ockham's razor and Bayesian analysis. *American Scientist*, 80:64–72, 1992.

[14] David J. C. MacKay. *Information Theory, Inference and Learning Algorithms*. 2003.

[15] Rémi Bardenet, Arnaud Doucet, and Chris Holmes. On markov chain monte carlo methods for tall data. *The Journal of Machine Learning Research*, 18(1):1515–1557, 2017.

[16] Christopher M Bishop. *Pattern recognition and machine learning*. Springer, 2006.

[17] Giorgio Parisi. *Statistical field theory*. Addison-Wesley, 1988.

[18] David M. Blei. Build, Compute, Critique, Repeat: Data Analysis with Latent Variable Models. *Annual Review of Statistics and Its Application*, 1(1):203–232, January 2014.

[19] David J. Lunn, Andrew Thomas, Nicky Best, and David Spiegelhalter. WinBUGS - A Bayesian modelling framework: Concepts, structure, and extensibility. *Statistics and Computing*, 10(4):325–337.

[20] Stan Development Team. Stan Modeling Language Users Guide and Reference Manual, Version 2.14.0. Technical report, 2016.

[21] T.P. Minka, J.M. Winn, J.P. Guiver, and D.A. Knowles. Infer.NET 2.4. Technical report, Microsoft Research, 2010.

[22] Noah Goodman, Vikash Mansinghka, Daniel M. Roy, Keith Bonawitz, and Joshua B. Tenenbaum. Church: a language for generative models. June 2012.

[23] Vikash Mansinghka, Daniel Selsam, and Yura Perov. Venture: a higher-order probabilistic programming platform with programmable inference. page 78, March 2014.

[24] Frank Wood, Jan Willem Meent, and Vikash Mansinghka. A new approach to probabilistic programming inference. In *Artificial Intelligence and Statistics*, pages 1024–1032, 2014.

[25] L. Li, Y. Wu, and S.J. Russell. SWIFT: Compiled Inference for Probabilistic Programs. Technical report, University of California, Berkley, 2015.

[26] Robert J Elliott, Lakhdar Aggoun, and John B Moore. *Hidden Markov models: estimation and control*, volume 29. Springer Science & Business Media, 2008.

[27] Arnaud Doucet, Simon Godsill, and Christophe Andrieu. On sequential monte carlo sampling methods for bayesian filtering. *Statistics and computing*, 10(3):197–208, 2000.

[28] Neil J Gordon, David J Salmond, and Adrian FM Smith. Novel approach to nonlinear/non-gaussian bayesian state estimation. In *IEE Proceedings F (Radar and Signal Processing)*, volume 140, pages 107–113. IET, 1993.

[29] Keiji Kanazawa, Daphne Koller, and Stuart Russell. Stochastic simulation algorithms for dynamic probabilistic networks. In *Proceedings of the Eleventh conference on Uncertainty in artificial intelligence*, pages 346–351. Morgan Kaufmann Publishers Inc., 1995.

[30] Michael Isard and Andrew Blake. A mixed-state condensation tracker with automatic model-switching. In *Computer Vision, 1998. Sixth International Conference on*, pages 107–112. IEEE, 1998.

[31] Christophe Andrieu, Arnaud Doucet, and Roman Holenstein. Particle markov chain monte carlo methods. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72(3):269–342, 2010.

[32] Dieter Fox, Sebastian Thrun, Wolfram Burgard, and Frank Dellaert. Particle Filters for Mobile Robot Localization. In *Sequential Monte Carlo Methods in Practice*, pages 401–428. Springer New York, New York, NY, 2001.

[33] Rudolph Van Der Merwe, N Ando De Freitas, and Eric Wan. The Unscented Particle Filter. *Advances in Neural Information Processing Systems*, pages 584–590, 2000.

[34] Yunpeng Li and Mark Coates. Particle Filtering with Invertible Particle Flow. jul 2016.

[35] Fred Daum, Jim Huang, and Arjang Noushin. Exact particle flow for nonlinear filters. In *SPIE Defense, Security, and Sensing*, pages 769704–769704. International Society for Optics and Photonics, 2010.

[36] Fred Daum and Jim Huang. Nonlinear filters with log-homotopy. In Oliver E. Drummond and Richard D. Teichgraeber, editors, *Proceedings of SPIE*, sep 2007.

[37] Muhammad Altamash Khan and Martin Ulmke. Non-linear and non-Gaussian state estimation using log-homotopy based particle flow filters. In *2014 Sensor Data Fusion: Trends, Solutions, Applications (SDF)*, pages 1–6. IEEE, oct 2014.

[38] Michael Montemerlo and Sebastian Thrun. FastSLAM 2.0. In *FastSLAM*, pages 63–90. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007.

[39] Augustine Kong, Jun S Liu, and Wing Hung Wong. Sequential Imputations and Bayesian Missing Data Problems. *Journal of the American Statistical Association*, 89(425):278–288, 1994.

[40] Jun S. Liu and Rong Chen. Sequential Monte Carlo Methods for Dynamic Systems. *Journal of the American Statistical Association*, 93(443):1032, sep 1998.

[41] Ming Lin, Rong Chen, and Jun S. Liu. Lookahead Strategies for Sequential Monte Carlo. *Statistical Science*, 28(1):69–94, feb 2013.

[42] Michael K Pitt and Neil Shephard. Filtering via simulation: Auxiliary particle filters. *Journal of the American statistical association*, 94(446):590–599, 1999.

[43] Ming T Lin, Junni L Zhang, Qiansheng Cheng, and Rong Chen. Independent Particle Filters. *Journal of the American Statistical Association*, 100(472):1412–1421, dec 2005.

[44] H. F. Trotter and J. W. Tukey. Conditional monte carlo for normal samples. In *Symposium on Monte Carlo Methods*, pages 64–79. John Wiley and Sons, 1956.

[45] Christophe Andrieu and Arnaud Doucet. Particle filtering for partially observed gaussian state space models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 64(4):827–836, 2002.

[46] Rong Chen and Jun S Liu. Mixture kalman filters. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 62(3):493–508, 2000.

[47] Arnaud Doucet, Nando De Freitas, Kevin Murphy, and Stuart Russell. Rao-blackwellised particle filtering for dynamic bayesian networks. In *Proceedings of the Sixteenth conference on Uncertainty in artificial intelligence*, pages 176–183. Morgan Kaufmann Publishers Inc., 2000.

[48] Thomas Schon, Fredrik Gustafsson, and P-J Nordlund. Marginalized particle filters for mixed linear/nonlinear state-space models. *IEEE Transactions on signal processing*, 53(7):2279–2289, 2005.

[49] Reinhart Kühne, Ralf-Peter Schäfer, Jürgen Mikat, and Stefan Lorkowski. New Approaches for Traffic Management in Metropolitan Areas. In *Proceedings of the 10th Symposium on Control in Transportation Systems*, Tokyo, 2003.

[50] Hector Gonzalez, Jiawei Han, Xiaolei Li, Margaret Myslinska, and John Paul Sondag. Adaptive fastest path computation on a road network: a traffic mining approach. In *Proceedings of the 33rd International Conference on Very Large Data Bases*, pages 794–805. VLDB Endowment, September 2007.

[51] Kira Kowalska, John Shawe-Taylor, and Paul Longley. Data-driven modelling of police route choice. In *Proceedings of the 23rd GIS Research UK conference*, 2015.

[52] Qingquan Li, Zhe Zeng, Tong Zhang, Jonathan Li, and Zhongheng Wu. Path-finding through flexible hierarchical road networks: An experiential approach using taxi trajectory data. *International Journal of Applied Earth Observation and Geoinformation*, 13(1):110–119, February 2011.

[53] Lin Liao, Donald J Patterson, Dieter Fox, and Henry Kautz. Building personal maps from GPS data. *Annals of the New York Academy of Sciences*, 1093:249–65, December 2006.

[54] Chris Brunsdon. Path estimation from gps tracks. In *Proceedings of the 9th International Conference on GeoComputation*. National Centre for Geocomputation, Maynooth University., 2007.

[55] Urška Demšar, Kevin Buchin, Francesca Cagnacci, Kamran Safi, Bettina Speckmann, Nico Van de Weghe, Daniel Weiskopf, and Robert Weibel. Analysis and visualisation of movement: an interdisciplinary review. *Movement ecology*, 3(1):5, 2015.

[56] David Barber. *Bayesian Reasoning and Machine Learning*. Cambridge University Press, 2012.

[57] Chong Yang Goh, Justin Dauwels, Nikola Mitrovic, Muhammad Tayyab Asif, Ali Oran, and Patrick Jaillet. Online map-matching based on hidden markov model for real-time traffic sensing applications. In *Intelligent Transportation Systems (ITSC), 2012 15th International IEEE Conference on*, pages 776–781. IEEE, 2012.

[58] George Rosario Jagadeesh and Thambipillai Srikanthan. Robust real-time route inference from sparse vehicle position data. In *Intelligent Transportation Systems (ITSC), 2014 IEEE 17th International Conference on*, pages 296–301. IEEE, 2014.

[59] Paul Newson and John Krumm. Hidden markov map matching through noise and sparseness. In *Proceedings of the 17th ACM SIGSPATIAL international conference on advances in geographic information systems*, pages 336–343. ACM, 2009.

[60] Sebastian Thrun. Particle filters in robotics. In *Proceedings of Uncertainty in AI*, pages 511–518. Morgan Kaufmann Publishers Inc., August 2002.

[61] Jun S Liu. *Monte Carlo strategies in scientific computing*. Springer Science & Business Media, 2008.

[62] Neil J. Gordon, Simon Maskell, and Thiagalingam Kirubarajan. Efficient particle filters for joint tracking and classification. In Oliver E. Drummond, editor, *Signal and Data Processing of Small Targets*, Orleando, 2002. International Society for Optics and Photonics.

[63] Sudarshan S. Chawathe. Segment-Based Map Matching. In *2007 IEEE Intelligent Vehicles Symposium*, pages 1190–1197. IEEE, jun 2007.

[64] Carola Wenk, Randall Salas, and Dieter Pfoser. Adressing the Need for Map-Matching Speed: Localizing Global Curve-Matching Algorithms. In *Pro-*

ceedings of the 18th international Conference on Scientific and Statistical Database Management, 2006.

[65] Helmut Alt, Alon Efrat, Günter Rote, and Carola Wenk. Matching planar maps. *Journal of Algorithms*, 49(2):262–283, 2003.

[66] Huabei Yin and O. Wolfson. A weight-based map matching method in moving objects databases. In *Proceedings. 16th International Conference on Scientific and Statistical Database Management, 2004.*, pages 437–438. IEEE.

[67] Oliver Pink and Britta Hummel. A statistical approach to map matching using road network geometry, topology and vehicular motion constraints. In *2008 11th International IEEE Conference on Intelligent Transportation Systems*, pages 862–867. IEEE, oct 2008.

[68] A Stewart Fotheringham. Trends in quantitative methods I: stressing the local. *Progress in Human Geography*, 21(1):88–96, 1997.

[69] Alexander D. Singleton and Paul A. Longley. Geodemographics, visualisation, and social networks in applied geography. *Applied Geography*, 29(3):289–298, jul 2009.

[70] Christian Thiemann, Fabian Theis, Daniel Grady, Rafael Brune, and Dirk Brockmann. The Structure of Borders in a Small World. *PLoS ONE*, 5(11):e15422, nov 2010.

[71] Filippo Simini, Marta C Gonzalez, Amos Maritan, and Albert-Laszlo Barabasi. A universal model for mobility and migration patterns. *Nature*, 484(7392):96–100, apr 2012.

[72] Vincent Blondel, Gautier Krings, Isabelle Thomas, and Jane Corrigan. Regions and borders of mobile telephony in Belgium and in the Brussels metropolitan zone. *Brussels Studies*, 42, 2010.

[73] Carlo Ratti, Stanislav Sobolevsky, Francesco Calabrese, Clio Andris, Jonathan Reades, Mauro Martino, Rob Claxton, and Steven H. Strogatz. Redrawing the Map of Great Britain from a Network of Human Interactions. *PLoS ONE*, 5(12):e14248, dec 2010.

[74] Dirk Brockmann. Following the money. *Physics World*, 23(02):31–34, feb 2010.

[75] Norbert Vanhove. *Regional policy: a European approach*. Ashgate Publishing Limited, Ashgate, 3 edition, 1999.

[76] Charlie Karlsson and Michael Olsson. The identification of functional regions: theory, methods, and applications. *The Annals of Regional Science*, 40(1):1–18, mar 2006.

[77] Ed Manley. Identifying functional urban regions within traffic flow. *Regional Studies, Regional Science*, 1(1):40–42, jan 2014.

[78] Elena Besussi, Nancy Chin, Michael Batty, and Paul Longley. The Structure and Form of Urban Settlements. In Tarek Rashed and Carsten Jürgens, editors, *Remote Sensing of Urban and Suburban Areas*, pages 13–31. Springer Netherlands, 2010.

[79] Paul Expert, Tim Evans, Vincent D. Blondel, and Renaud Lambiotte. Uncovering space-independent communities in spatial networks. *Proceedings of the National Academy of Sciences of the United States of America*, 108(19):7663–7668, dec 2011.

[80] Ordnance Survey. OS MasterMap Integrated Transport Network Layer, 2017.

[81] M. E. J. (Mark E. J.) Newman. *Networks : an introduction*. Oxford University Press, 2010.

[82] R Guimerà, S Mossa, A Turtschi, and L A N Amaral. The worldwide air transportation network: Anomalous centrality, community structure, and

cities' global roles. *Proceedings of the National Academy of Sciences of the United States of America*, 102(22):7794–9, may 2005.

[83] Jukka-Pekka Onnela, Samuel Arbesman, Marta C. González, Albert-László Barabási, and Nicholas A. Christakis. Geographic Constraints on Social Network Groups. *PLoS ONE*, 6(4):e16939, apr 2011.

[84] M. Newman and M. Girvan. Finding and evaluating community structure in networks. *Physical Review E*, 69(2):026113, feb 2004.

[85] Ulrik Brandes, Daniel Delling, Marco Gaertler, Robert Görke, Martin Hoefer, Zoran Nikoloski, and Dorothea Wagner. *On Finding Graph Clusterings with Maximum Modularity*, pages 121–132. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007.

[86] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10), oct 2008.

[87] Lucas G. S. Jeub, Marya Bazzi, Inderjit S. Jutla, and Peter J. Mucha. A generalized Louvain method for community detection implemented in MATLAB, 2016.

[88] David M. Blei. Probabilistic topic models. *Communications of the ACM*, 55(4):77, apr 2012.

[89] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

[90] Thomas L Griffiths and Mark Steyvers. Finding scientific topics. *Proceedings of the National Academy of Sciences of the United States of America*, 101(1):5228–35, apr 2004.

[91] Nicola Barbieri, Giuseppe Manco, Ettore Ritacco, Marco Carnuccio, and Antonio Bevacqua. Probabilistic topic models for sequence data. *Machine Learning*, 93(1):5–29, jul 2013.

[92] Matthew D. Homan and Andrew Gelman. The No-U-turn sampler: adaptively setting path lengths in Hamiltonian Monte Carlo. *The Journal of Machine Learning Research*, 15(1):1593–1623, jan 2014.

[93] Huanfa Chen, Tao Cheng, and Sarah Wise. Developing an online cooperative police patrol routing strategy. *Computers, Environment and Urban Systems*, 62:19–29, 2017.

[94] Anthony A. Braga, Andrew V. Papachristos, and David M. Hureau. The Effects of Hot Spots Policing on Crime: An Updated Systematic Review and Meta-Analysis. *Justice Quarterly*, 31(4):633–663, jul 2014.

[95] Juuso Parkkinen, Adam Gyenge, and Janne Sinkkonen. A block model suitable for sparse graphs. *In Proceedings of the 7th International Workshop on Mining and Learning with Graphs*, 2009.

[96] Nguyen Xuan Vinh, Unsweduau Julien Epps, and James Bailey. Information Theoretic Measures for Clusterings Comparison: Variants, Properties, Normalization and Correction for Chance. *Journal of Machine Learning Research*, 11:2837–2854, 2010.

[97] Jan M Wiener. 'Fine-to-Coarse Route Planning and Navigation in Regionalized Environments. *Journal of Spatial Cognition and Computation*, 3(4):331–358, 2003.

[98] E.J. Manley, S.W. Orr, and T. Cheng. A heuristic model of bounded route choice in urban areas. *Transportation Research Part C: Emerging Technologies*, 56:195–209, jul 2015.

[99] Tâm Huynh, Mario Fritz, and Bernt Schiele. Discovery of activity patterns using topic models. In *Proceedings of the 10th international conference on Ubiquitous computing*, pages 10–19. ACM, 2008.

[100] Katayoun Farrahi and Daniel Gatica-Perez. Extracting mobile behavioral patterns with the distant n-gram topic model. In *Wearable Computers (ISWC), 2012 16th International Symposium on*, pages 1–8. IEEE, 2012.

[101] Agresti Alan and Finlay Barbara. Statistical methods for the social sciences, 2009.

[102] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[103] Fredrik Lindsten, Adam M Johansen, Christian A Naesseth, Bonnie Kirkpatrick, Thomas B Schön, JAD Aston, and Alexandre Bouchard-Côté. Divide-and-conquer with sequential monte carlo. *Journal of Computational and Graphical Statistics*, 26(2):445–458, 2017.

[104] Adrien Todeschini, François Caron, Marc Fuentes, Pierrick Legrand, and Pierre Del Moral. Biips: software for bayesian inference with interacting particle systems. *arXiv preprint arXiv:1412.3779*, 2014.

[105] Brooks Paige and Frank Wood. Inference networks for sequential monte carlo in graphical models. In *International Conference on Machine Learning*, pages 3040–3049, 2016.

[106] Shixiang Gu, Zoubin Ghahramani, and Richard E Turner. Neural adaptive sequential monte carlo. In *Advances in Neural Information Processing Systems*, pages 2629–2637, 2015.

[107] Shakir Mohamed and Balaji Lakshminarayanan. Learning in implicit generative models. *arXiv preprint arXiv:1610.03483*, 2016.

[108] Ferenc Huszár. Variational inference using implicit distributions. *arXiv preprint arXiv:1702.08235*, 2017.

[109] Lars Mescheder, Sebastian Nowozin, and Andreas Geiger. Adversarial variational bayes: Unifying variational autoencoders and generative adversarial networks. *arXiv preprint arXiv:1701.04722*, 2017.

[110] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[111] Arnaud Doucet and Am Johansen. A tutorial on particle filtering and smoothing: fifteen years later. *Handbook of Nonlinear Filtering*, 12(December):656–704, 2009.

[112] Arnaud Doucet, Nando De Freitas, and NJ Gordon. Sequential monte carlo methods in practice. series statistics for engineering and information science, 2001.

[113] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[114] Kevin Roth, Aurelien Lucchi, Sebastian Nowozin, and Thomas Hofmann. Stabilizing training of generative adversarial networks through regularization. In *Advances in Neural Information Processing Systems*, pages 2015–2025, 2017.

[115] Chun-Liang Li, Wei-Cheng Chang, Yu Cheng, Yiming Yang, and Barnabás Póczos. Mmd gan: Towards deeper understanding of moment matching network. In *Advances in Neural Information Processing Systems*, pages 2200–2210, 2017.

[116] Yujia Li, Kevin Swersky, and Rich Zemel. Generative moment matching networks. In *International Conference on Machine Learning*, pages 1718–1727, 2015.

[117] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082*, 2014.

[118] Anders Boesen Lindbo Larsen, Søren Kaae Sønderby, Hugo Larochelle, and Ole Winther. Autoencoding beyond pixels using a learned similarity metric. *arXiv preprint arXiv:1512.09300*, 2015.

[119] Danilo Jimenez Rezende and Shakir Mohamed. Variational inference with normalizing flows. *arXiv preprint arXiv:1505.05770*, 2015.

[120] Bharath K Sriperumbudur, Kenji Fukumizu, Arthur Gretton, Bernhard Schölkopf, and Gert RG Lanckriet. On integral probability metrics,\phi-divergences and binary classification. *arXiv preprint arXiv:0901.2698*, 2009.

[121] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

[122] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[123] Radford M Neal. Annealed importance sampling. *Statistics and computing*, 11(2):125–139, 2001.

[124] Diederik P Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. Improved variational inference with inverse autoregressive flow. In *Advances in Neural Information Processing Systems*, pages 4743–4751, 2016.

[125] Lars Maaløe, Casper Kaae Sønderby, Søren Kaae Sønderby, and Ole Winther. Auxiliary deep generative models. *arXiv preprint arXiv:1602.05473*, 2016.

[126] Yuhuai Wu, Yuri Burda, Ruslan Salakhutdinov, and Roger Grosse. On the quantitative analysis of decoder-based generative models. *arXiv preprint arXiv:1611.04273*, 2016.

[127] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. In *Advances in Neural Information Processing Systems*, pages 2234–2242, 2016.

[128] Christopher M Bishop. Model-based machine learning. *Philosophical transactions. Series A, Mathematical, physical, and engineering sciences*, 371(1984):20120222, February 2013.

[129] Emanuel Parzen. On estimation of a probability density function and mode. *The annals of mathematical statistics*, 33(3):1065–1076, 1962.

[130] Lucas Theis, Aäron van den Oord, and Matthias Bethge. A note on the evaluation of generative models. *arXiv preprint arXiv:1511.01844*, 2015.

[131] Kira Kempinska, Toby Davies, John Shawe-Taylor, and Paul Longley. Probabilistic map-matching for low-frequency gps trajectories. In *Proceedings of GIS Ostrava conference*, pages 209–221. Springer, 2017.

[132] Kira Kempinska, Toby O. Davies, and John Shawe-Taylor. Probabilistic map-matching using particle filters. In *Proceedings of the 24th GIS Research UK conference*, 2016.

[133] Kira Kempinska and John Shawe-Taylor. Improved particle filters for vehicle localisation. In *NIPS 2016 Advances in Approximate Inference workshop*, 2016.

[134] Kira Kempinska, Paul Longley, and John Shawe-Taylor. Interactional regions in cities: making sense of flows across networked systems. *International Journal of Geographical Information Science*, 32(7):1348–1367, 2018.

[135] T Cheng, Kate Bowers, Paul Longley, John Shawe-Taylor, Trevor Adams, Toby Davies, Gabriel Rosser, Sarah Wise, Chris Gale, Monsuru Adepeju, Jianan Shen, Huanfa Chen, Dawn Williams, Kira Kempinska, and Artemis Skarlatidou. *CPC: Crime, Policing and Citizenship - Intelligent policing and big data.* 05 2016.

[136] Kira Kempinska and John Shawe-Taylor. Adversarial sequential monte carlo. In *NIPS 2017 Advances in Approximate Inference workshop*, 2017.

.