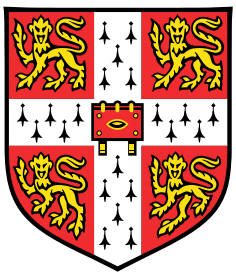
Stochastic Modelling and Approximate Bayesian Inference: Applications in Object Tracking and Intent Analysis



Runze Gan

Department of Engineering
University of Cambridge

This dissertation is submitted for the degree of
Doctor of Philosophy

Sidney Sussex College

April 2023

To my loving parents and family

Declaration

This thesis is the result of my own work and includes nothing which is the outcome of work done in collaboration except as declared in the Preface and specified in the text. I further state that no substantial part of my thesis has already been submitted, or, is being concurrently submitted for any such degree, diploma or other qualification at the University of Cambridge or any other University or similar institution except as declared in the Preface and specified in the text. It does not exceed the prescribed word limit for the relevant Degree Committee.

Runze Gan
April 2023

Abstract

Stochastic Modelling and Approximate Bayesian Inference: Applications in Object Tracking and Intent Analysis

Runze Gan

As two fundamental pillars of Bayesian inference for time series, stochastic modelling and approximate Bayesian inference play crucial roles in providing accurate priors for underlying random processes and addressing the challenges of evaluating posterior distributions when exact computation is infeasible. Balancing novel contributions in both areas, this thesis highlights innovative stochastic models in Chapters 2 and 3, and puts emphasis on novel approximate Bayesian inference schemes in Chapters 4, 5, and 6. Driven by applications in object tracking and intent inference, the developed methodologies aim to accurately capture desired motion characteristics while enhancing the effectiveness, efficiency, and robustness of estimations.

The applications of intent inference and single object tracking are considered in Chapters 2 and 3. Chapter 2 presents a generic Bayesian intent inference framework capable of predicting the destination of a tracked object, along with an exploration of several mean-reverting stochastic processes that serve as dynamic models within the framework. Chapter 3 develops novel α-stable Lévy state-space models for manoeuvring object tracking and intent prediction, expressed in continuous time as Lévy processes. These models effectively capture sharp changes in state induced by erratic maneuvers with heavy-tailed α-stable driven noise, while maintaining an advantageous conditionally Gaussian transition. Additionally, this chapter introduces an efficient intent inference procedure that accommodates dynamically varying intent across the surveyed area, offering versatile solutions for diverse tracking scenarios.

Chapter 4 introduces a novel conditionally factorised variational family that retains dependence between desired variables at user-defined levels of detail. A new variational Bayes algorithm is then proposed and implemented with importance sampling. It

guarantees a better variational lower bound by choosing a finer conditional structure, offering a flexible trade-off between computational cost and inference accuracy.

Multi-object tracking tasks are addressed in Chapters 5 and 6 with Poisson measurement processes. Chapter 5 introduces a variational Bayes multi-object tracker that effectively performs tracking, data association, and learning of target and clutter rates, while offering substantial efficiency gains and parallelisable implementation. Chapter 6 extends this tracker to tackle highly challenging tracking scenarios involving a large number of closely-spaced objects and heavy clutter. By introducing a novel variational localisation strategy that quickly rediscovers missed targets under extremely heavy clutter, the enhanced tracker can automatically detect and recover from track loss, delivering outstanding performance in tracking accuracy and efficiency under difficult tracking conditions.

Acknowledgements

I would like to express my sincere gratitude to my supervisor, Simon, for his exceptional guidance, valuable insights, and sustained support throughout my PhD studies. I am very grateful for his patient guidance in the early stages of my PhD studies, providing hands-on help with even the most trivial tasks; and for his open-minded supervision in the later stages, when he encouraged me to pursue my research interests while placing his trust in my work and the direction I chose. These experiences have nurtured my growth as a more independent, resilient, and proficient researcher, while also developing my own academic taste and research interests, which have undoubtedly been influenced by him.

I would also like to express my gratitude to Jiaming and Bashar for their valuable support and collaboration during the early stages of my PhD, both of whom have been excellent colleagues and friends. Their discussions and feedback on our intent inference paper, have been crucial in shaping my work on this topic. I am also thankful to Rachel for her administrative support throughout my PhD study.

I am thankful to my soulmate, Lily, for her unwavering support and belief in my abilities throughout this journey. Her collaboration on research projects, engaging discussions on relevant literature, constructive feedback on my writing, and encouragement during difficult times have been indispensable and a continuous source of inspiration. I am fortunate to have her by my side on this journey, which has become much more enjoyable and meaningful as a result. It is a privilege to have her in my life, and I feel deeply grateful for the love, support, and companionship she has brought into my life.

Lastly, I would like to express my heartfelt gratitude to my parents, who have given me unconditional love throughout my life. Through their guidance in my early years, I developed the invaluable quality of persevering through tough problems, even when they are frustrating and require a significant investment of time. They supported me in pursuing any research I was passionate about, and encouraged me to chase my dreams without fear, for which I am eternally grateful. Thanks to them, I have been able to

complete this complex, time-consuming, and fascinating piece of work, driven solely by my genuine interest and curiosity rather than external rewards or recognition.

Table of contents

Nomenclature

Acronyms / Abbreviations

AbNHPP Association-based Non-homogeneous Poisson Process

BD Bridging Distribution

CA Constant Acceleration

CAVI Coordinate Ascent Variational Inference

CV Constant Velocity

CVB Conditionally Factorised Variational Bayes

DLD Dynamic Latent Destination

ELBO Evidence Lower Bound

EM Expectation–Maximisation

ERA Equilibrium Reverting Acceleration

ERV Equilibrium Reverting Velocity

ET-JPDA Extended Target Joint Probabilistic Data Association

GUI Graphical User Interface

HCI Human Computer Interaction

IS Importance Sampling

IS-CVB Importance Sampling based Conditionally Factorised Variational Bayes

KL divergence Kullback-Leibler divergence

MCMC Markov Chain Monte Carlo

MRD Mean Reverting Diffusion

NHPP Non-homogeneous Poisson Process

ODE Ordinary Differential Equation

OSPA Optimal Sub Pattern Assignment

OU Ornstein-Uhlenbeck

PED Prediction Error Decomposition

PHD Probabilistic Hypothesis Density

PMBM Poisson Multiple Bernoulli Mixture

PMHT Probabilistic Multiple Hypothesis Tracker

RFS Random Finite Set

RMSE Root Mean Square Error

SCFG Stochastic Context-free Grammar

SDE Stochastic Differential Equation

SMCMC Sequential Markov Chain Monte Carlo

SPRT Sequential Probability Ratio Test

VB-AbNHPP Variational Bayes Association-based NHPP

VB-AbNHPP-RELO VB-AbNHPP Tracker with Relocation

Notations / Symbols

Notations and symbols utilised in this thesis are defined individually for each chapter, with the exception that Chapters 5 and 6 share the same notations. It should be noted that bold symbols are exclusively used in Chapters 2 and 3 to represent multivariate variables.

Chapter 1

Introduction

Bayesian inference [1] offers a coherent framework for incorporating prior knowledge and updating it in light of new information, with broad applicability across diverse disciplines such as science, philosophy, and engineering. At the heart of this framework lies Bayes' theorem. Independently developed in the 18th century by Thomas Bayes [2] and Pierre-Simon Laplace [3], this theorem has since inspired a wealth of research efforts, especially in recent decades. The work of this thesis builds upon this rich tradition. Expressed in modern notation, Bayes' theorem allows for the computation of the posterior distribution of a random variable X given an observed variable Y:

$$p(X|Y) = \frac{p(Y|X)p(X)}{p(Y)}, \tag{1.1}$$

where $p(X)$ is the prior distribution that represents our initial knowledge of X, and the conditional distribution $p(Y|X)$ is the likelihood, which indicates how probable it is for different values of X to produce the observed Y. The posterior distribution $p(X|Y)$ reflects our new understanding of X after taking the observed data into account. This elegant formula ensures a mathematically consistent and systematic updating of prior knowledge with new evidence.

One of the most appealing characteristics of Bayesian inference is its recursive nature: when new observations become available, the posterior from the previous time step may be used to build the new prior, enabling a continuous refinement of beliefs in an online fashion. This recursive Bayesian inference mirrors the natural progression from prior knowledge to updated knowledge as evidence accumulates over time. Consequently, it underpins many sequential inference tasks, such as multi-object tracking [4, 5] and online intent inference [6, 7] — the primary applications considered in this thesis.

To successfully implement Bayesian inference using Bayes' rule in (1.1), two key aspects must be addressed: 1) the proper definition of the prior distribution $p(X)$, which includes specifying the underlying stochastic process when X represents a collection of time series variables; and 2) the evaluation of the posterior $p(X|Y)$ according to (1.1), which may require approximation techniques when direct computation is intractable. These aspects correspond respectively to the primary theoretical and methodological focus of this thesis, namely stochastic modelling and approximate Bayesian inference.

Stochastic modelling aims to provide an accurate prior for the underlying random process, capturing the inherent uncertainty and variability of the interested real-world time series. This is particularly important for our considered application of object tracking, where typically only limited amounts of data, possibly of unsatisfactory quality, are received at each time step. Despite these challenges, reliable estimations are required at every stage, making an accurate prior and, therefore, effective stochastic modelling essential. This thesis addresses the stochastic modelling problem by developing meaningful stochastic processes [8, 9] that match the desired characteristics of the real-world time series, as informed by physical laws or empirical observations. We will specifically focus on the stochastic modelling for object tracking and/or intent inference, for which the constructed stochastic process serves as the motion/dynamic model [10, 4] of the tracked object.

Approximate Bayesian inference addresses the challenge of evaluating the posterior distribution $p(X|Y)$ when exact computation in (1.1) is not feasible. This research area has gained more attention compared to stochastic modelling, as it is not only relevant for sequential inference tasks but also for non-time series based machine learning tasks. We refer to [11] for a concise recent review of such methods and [1, 12] for more in-depth descriptions of specific approaches. In this thesis, we concentrate on two popular approximate inference techniques: Monte Carlo methods and variational inference. When dealing with tracking applications, we place emphasis on their sequential implementation.

Besides the aforementioned components, a comprehensive Bayesian inference algorithm also requires the definition of the likelihood function $p(Y|X)$ in (1.1). For object tracking, this function emerges from the sensor characteristics related to a specific task, and its parametric form is often well-known due to the extensive development in the tracking field for more than 50 years. While this thesis does not introduce new likelihood functions (though it offers new interpretations of existing ones, as in Section 5.2), it emphasises the development of stochastic modelling and

approximate Bayesian inference, both of which are vital ingredients for conducting recursive/sequential Bayesian inference for the considered tasks.

1.1 Background

While detailed introductions to the primary applications of this thesis will be provided in later chapters, including intent inference in Sections 2.1 and 3.1, as well as multi-object tracking in Sections 5.1 and 6.1, we offer a concise review from the methodology perspective in this section. This review will focus on stochastic modelling and approximate Bayesian inference, encompassing the techniques utilised throughout this thesis.

1.1.1 Stochastic modelling

This thesis utilises stochastic differential equations (SDEs) [13, 14] for the stochastic modelling of the considered time series, offering several advantages. SDEs represent the continuous-time evolution of a system or process influenced by both deterministic and stochastic factors. The deterministic factor captures the overall trend of a process, which allows for the integration of prior knowledge regarding the movement patterns of the time series of interest. On the other hand, the stochastic factor naturally models the uncertainty inherent in the evolution of the time series. Solving an SDE yields a continuous-time and, in most cases, Markovian transition density. The continuous-time property aligns with real-world movement in tracking applications, enabling more accurate representations of the underlying process and estimation with asynchronous measurements or inference between observation times. The Markovian characteristic greatly simplifies the filtering process and improves computational efficiency. Overall, employing SDEs for stochastic modelling offers a coherent and effective approach to capture the dynamics and uncertainties of interested time series for Bayesian inference.

Multi-dimensional SDEs are often required in tracking applications to model not just position, but also higher-order dynamics such as velocity and heading. Well-known explicit solutions for multi-dimensional SDEs are restricted to linear cases, as detailed in Section 4.3 of [14]. Scalar SDEs have slightly more general explicit solutions, as outlined in Section 5.3 of [15]. Additional solvable SDEs can be found in [9]. For general SDEs capturing desired characteristics of time series, numerical solutions or approximations may be necessary, as covered in [14, 9]. Notably, exact simulation

can be applied to more general SDE types [16, 17], facilitating accurate Monte Carlo inference methods.

While the aforementioned (approximate) solution and exact simulation techniques apply primarily to SDEs driven by Brownian motion, valuable SDEs can be driven by more general Lévy processes [18]. See [19, 20] for the simulation of specific types of Lévy processes, and [21] for the study of the SDEs they drive. In Chapter 3, we will develop a model using one such SDE for tracking and intent inference applications.

In addition to creating entirely new SDEs for modelling the time series of interest, new stochastic processes can be constructed by transforming known Markov processes. Such transformations include conditioning, killing, time changes, state space transformations, and others. See [8, 22] for introductions to these techniques. In Chapter 2, we will construct a new model using conditioning techniques to transform existing widely-adopted processes for intent inference applications.

1.1.2 Approximate Bayesian inference

It is often the case that exact inference for posterior $p(X|Y)$ in (1.1) is analytically infeasible, in which case approximate inference techniques can be used to handle the approximation of posterior distribution. An important class of such approximation inference approaches is variational inference [23], which seek to approximate the true posterior distribution with a simpler, more tractable distribution. By transforming the inference problem into an optimisation problem, it seeks to minimise the Kullback-Leibler (KL) divergence (which measures the difference between two distributions) between the true and approximated distributions. Such variational methods can handle large data, and are demonstrated to be efficient in the tracking tasks considered in this thesis, which will be detailed in Chapters 5 and 6.

Another important inference method is the probabilistic graphical model [24, 12, 25]. Graphical model approaches provide a simple way to visualise the structure of a probabilistic model and to design inference algorithm by exploiting these graphical structures [12]. For exact inference problems such as tree-structured graphs, the sum-product and max-sum algorithms provide efficient solutions with local message passing around the graph. However, for graphs with loops, approximate inference, known as loopy belief propagation, was proposed with message passing rules [26]. However, due to the cycles in the graph, the convergence of loopy belief propagation is not guaranteed; In some applications, the algorithm will converge, whereas for others it will not. For a detailed discussion, please see e.g. [26, 12].

Besides these deterministic approaches, a more flexible and versatile scheme is the Monte Carlo method [27], which is based on stochastic numerical sampling from distributions. Monte Carlo sampling methods have the capacity to produce samples from intricate, high-dimensional distributions, providing a more accurate representation of the target distribution. However, they come with their own set of challenges, notably the computational complexity and potential for slow convergence.

This thesis addresses the intractable posterior $p(X|Y)$ in (1.1) by employing Monte Carlo methods [27] and variational inference [23, 28], the two predominant approximate inference techniques. In this section, we first provide a brief review of the fundamental concepts of variational inference and Monte Carlo methods in the context of non-sequential inference. Subsequently, we discuss sequential approximate inference, also known as approximate filtering.

1.1.2.1 Variational inference

While the exact posterior $p(X|Y)$ is intractable, variational inference aims to approximate it by a variational distribution $q(X)$ that minimises its KL divergence to the exact $p(X|Y)$, i.e. $\mathrm{KL}(q(X)||p(X|Y))$. However, without any restrictions on $q(X)$, the optimal variational distribution would be the exact posterior itself, which is not helpful due to its intractability. Consequently, when implementing variational inference, an initial step involves defining a variational family (encompassing all variational distributions to select from). From this family, a member is then selected that minimises the $\mathrm{KL}(q(X)||p(X|Y))$, providing a more tractable approximation. This process transforms the intractable Bayesian inference problem into an optimisation problem, allowing for the application of highly effective optimisation techniques. In the implementation of variational inference, minimising $\mathrm{KL}(q(X)||p(X|Y))$ is replaced by an equivalent optimisation task: minimising the more manageable quantity known as evidence lower bound (ELBO). More details can be found in [23] and Section 4.1.1 of this thesis.

A classical method in variational inference is coordinate ascent variational inference (CAVI) [23, 12], also known as mean-field variational inference. CAVI assumes a mean-field variational family and utilises the coordinate ascent algorithm to optimise the ELBO. This method will be the primary focus of the thesis, with a more detailed review of its properties and extensions in Sections 4.1.2. Over the past decade, several modern variational inference algorithms have emerged to address the limitations of classical CAVI in machine learning tasks. For instance, CAVI can be computationally expensive for large datasets because each update iteration requires processing the entire dataset. Stochastic variational inference [29] addresses this by leveraging stochastic optimisation

and natural gradient methods, enabling updates with single data points or mini-batches, and thus scaling CAVI to massive datasets. Additionally, classical CAVI does not guarantee a well-known parametric form for the variational distribution and requires laborious model-specific derivations. Black-box variational inference [30] overcomes these issues by using stochastic gradient descent updates with a predefined parametric form of the variational distribution. Similarly, the popular variational autoencoder [31] addresses these limitations by employing an alternative type of stochastic gradient for training neural networks. For other recent advancements of variational inference, see [28].

Despite the aforementioned disadvantages of classical CAVI, it is important to note that these drawbacks are generally not significant concerns for the signal processing tasks considered in this thesis, such as object tracking. In these tasks, the data received at each time step is usually not large, and the emphasis is placed on implementing sequential inference. Furthermore, for the tracking tasks examined in this thesis, the standard deterministic coordinate ascent variational update can yield well-known parametric forms. As a result, the stochastic gradient descent employed in modern variational inference to maintain the desired parametric form becomes unnecessary. More importantly, whilst the stochastic gradient aims to approximate the exact gradient, it nonetheless introduces random perturbations into the exact value. This can result in suboptimal performance or slower convergence rates compared to exact gradient descent or other deterministic optimisation used in classical CAVI. Additional disadvantages of using gradient-based variational inference for more general inference tasks are discussed in Section 4.1.3. Consequently, this thesis focuses on further developing CAVI for object tracking and general signal processing tasks, rather than adopting the aforementioned modern variational inference methods. Nonetheless, these methods offer valuable directions for future exploration, depending on different model assumptions or task requirements.

1.1.2.2 Monte Carlo methods

Although the exact posterior $p(X|Y)$ is intractable, Monte Carlo methods [27, 12] aim to generate a large number of samples (say, N_p) $X^{(p)}$ $(p = 1, 2, ..., N_p)$ from $p(X|Y)$. These samples can then be used to form an empirical approximation of $p(X|Y)$ through histograms, kernel density estimation, or simply by employing $\frac{1}{N_p}\sum_{p=1}^{N_p}\delta(X^{(p)})$, where $\delta(\cdot)$ represents the Dirac Delta function, provided the samples are independent. A more impactful application of Monte Carlo methods is evaluating the expectation of a desired function $\mathrm{E}_{p(X|Y)}f(X)$. In many cases, this expectation is the primary requirement

for the considered inference task, while the exact form of the posterior $p(X|Y)$ is less critical. Monte Carlo methods provide a straightforward and unbiased estimate of this quantity, denoted as $\hat{f} = \sum_{p=1}^{N_p} f(X^{(p)})$. Most importantly, when samples $X^{(p)}$ are independent, the variance of this estimate scales with $\frac{1}{N_p}$ [27, 12] and hence converges to 0 as N_p becomes large. This implies that the estimate $\hat{f}$ can approach the ground truth with an arbitrary level of accuracy, provided the sample size is large enough. This asymptotic property distinguishes Monte Carlo methods from variational inference, where estimates derived from the variational distribution generally cannot guarantee arbitrary accuracy.

However, this asymptotic property also suggests that Monte Carlo methods may be slow, as a large number of samples are needed to achieve a small variance and, consequently, an accurate estimate. Various techniques have been proposed to reduce the variance of these estimates, including Rao-Blackwellisation, control variates, and coupling random numbers/antithetic variables, among others. We refer readers to Chapter 4 of [27] for more details. In this thesis, we will employ Rao-Blackwellisation in a sequential Monte Carlo setting, which will be discussed later.

Although beneficial, implementing the simple Monte Carlo method described above is generally challenging for most inference tasks, as directly generating samples from the intractable posterior $p(X|Y)$ is often difficult. However, in most cases, we can obtain an unnormalised posterior using the prior and likelihood in (1.1). Leveraging this unnormalised posterior, various clever algorithms have emerged throughout the history of Monte Carlo methods. Examples of these methods include rejection sampling, which generates exact samples but can be inefficient; importance sampling, which estimates the integral $\mathrm{E}_{p(X|Y)} f(X)$ but is often ineffective for high-dimensional variables X and exhibits slight bias when only the unnormalised posterior, rather than the exact posterior, is known (see e.g. Section 23.4.2 in [32]), though its implementation is efficient; and Markov chain Monte Carlo (MCMC), which constructs a Markov chain targeting the distribution $p(X|Y)$ and allows for approximately independent samples once the chain has converged, but it can be slow. In addition to these methods, there are many other sampling techniques, each with their own strengths and weaknesses. These methods have numerous variants, with MCMC, in particular, encompassing a wide range of ingenious algorithms (see [33]).

Despite the aforementioned drawbacks, importance sampling remains one of the most straightforward and efficient sampling techniques to implement. Moreover, challenges related to the curse of dimensionality can be alleviated through the utilisation of improved proposal distributions or by adopting sequential implementations featuring a

sequence of intermediate target distributions [34, 35]. In this thesis, our primary focus will be on importance sampling, owing to its efficiency. This method will exclusively be applied to low-dimensional variables, all of which have fewer than four dimensions throughout the thesis.

1.1.2.3 Approximate sequential Bayesian inference

In sequential Bayesian inference or filtering processes [36], the exact posterior from the previous time step is used to construct the new prior of the current time step according to the transition density. This further challenges the tractability of the inference result, as the size of the parameters that sufficiently define the posterior and the computational cost required for each time step should not increase as the process evolves over time to avoid escalating memory and computational cost requirements for filtering. Aside from filtering process for hidden Markov models, which can be handled with the forward-backward algorithm [37, 12] (requiring only the forward pass), and linear Gaussian systems that can be addressed by the Kalman filter [38], sequential (Bayesian) inference of general state-space models necessitates approximation. However, it is worth noting that the Kalman filter can still produce the best linear estimator (in minimising the mean square error senses) for linear non-Gaussian systems in a sequential manner [39].

In the context of approximate filtering, a widely adopted intuitive approach, which we refer to as empirical filtering, is used by many algorithms. This method involves approximating the exact posterior with a simpler distribution and directly propagating it to the next time step to construct a new prior and continue evaluating the posterior. Although simple, this approach introduces new approximation errors at each time step. These errors can be difficult to quantify, and there is a lack of literature addressing this issue for extended Kalman filters [36], unscented Kalman filters [40], and other general Gaussian filters [36]. In contrast, importance sampling particle filters have established many sharp convergence results, with particularly strong results for models possessing the exponential forgetting property; see [41–43] for details. Also, see [44] for the theoretical support for sequential MCMC methods. These theoretical justifications make sequential Monte Carlo methods more rigorous than other empirical filtering algorithms. In this thesis, Rao-Blackwellised particle filters [45–47] will be utilised in the first two chapters. See also the overviews of general importance sampling particle filters [48, 43, 46]. Rao-Blackwellisation in these sequential Monte Carlo settings not only reduces estimate variance [49] but also decreases the sampling dimension,

significantly improving performance for multi-object tracking tasks where the sampling variable, comprising all target states, has a large dimension [50].

Numerous works have implemented variational inference sequentially as approximate filtering using the empirical filtering framework described earlier, such as in [51–53]. However, these methods are limited to specific models or a small class of models. In Section 5.3, we will propose an approach for more general dynamic systems. Recently, some research has proposed variational filtering methods that, instead of using the empirical filtering framework, focus on optimising a KL divergence for the joint posterior across all time steps in a sequential manner; see [54, 55]. These methods exhibit an elegant structure as they treat the same objective over different time steps, with the optimisation at each time step contributing incremental improvements to the objective. However, the implementation of these methods relies on stochastic gradient descent for optimisation and currently only applies to simple state space models (with only one sequential variable to infer per time step, but potentially with large dimensions). Moreover, it remains unclear whether these methods outperform simple empirical filtering in terms of mean squared error, which would be an interesting topic for future research.

Although this thesis primarily focuses on the Bayesian filtering task, it is worth noting that other signal processing problems, such as smoothing [36, 56–58] and parameter learning [41, 59, 60], also present interesting avenues for future exploration.

1.2 Thesis outline and main contributions

As two fundamental pillars of Bayesian inference, stochastic modelling and approximate Bayesian inference each encompass vast research fields with extensive studies conducted in each area. Recognising the impracticality of examining every facet of these extensive fields, this thesis endeavors to strike a balance by making novel contributions in both areas. In particular, **Chapters 2** and **3** emphasise the development of innovative stochastic models, while **Chapters 4**, **5** and **6** mainly contribute novel approximate Bayesian inference schemes.

The methodologies developed in the thesis are primarily driven by applications in object tracking and intent inference, aiming to capture desired motion characteristics while enhancing the efficacy, efficiency, and robustness of the estimations. In particular, the intent inference and single object tracking applications are considered in **Chapters 2** and **3**, while **Chapters 5** and **6** focus on multi-object tracking tasks.

In the following, we outline the research focus and primary contribution of each chapter. A more comprehensive list of contributions can be found in the 'Contributions' section of each chapter. Moreover, **Chapter 7** systematically highlights the contributions, categorising them according to their relevance to stochastic modelling, approximate inference, and application aspects.

Beginning with a motivating human computer interaction (HCI) example of predictive touch, **Chapter 2** presents a generic Bayesian intent inference framework capable of predicting the destination of a tracked object from a finite number of nominal endpoints. This chapter establishes the fundamental assumptions for the online intent inference task considered in this thesis and explains the rationale behind the presented framework. In the context of stochastic modelling, the main contribution is an exploration of several mean-reverting stochastic processes that can be employed as dynamic models within the presented framework. Notably, one of the presented models achieves the highest intent inference accuracy in the considered predictive touch application to date.

In **Chapter 3**, we develop novel α-stable Lévy state-space models for manoeuvring object tracking and/or intent prediction, expressed in continuous time as Lévy processes. In contrast to conventional (fully) Gaussian formulations, the proposed models are driven by heavy-tailed α-stable noise and are thus much more able to capture extreme values/behaviours. This can better characterise sharp changes in the state, which may be induced by sudden and frequent manoeuvres such as swift turns or abrupt accelerations. The proposed models are constructed in a conditionally Gaussian form, allowing for efficient sequential inference using a Rao-Blackwellised particle filter. With great flexibility, the proposed stochastic modelling and inference framework can be easily adapted to many widely adopted continuous-time dynamic models by replacing Gaussian-driven noise with α-stable noise. Moreover, we introduce an efficient intent inference procedure that accommodates a dynamically changing or static intended endpoint anywhere within the surveyed area, unlike the approach taken in **Chapter 2**.

In previous chapters, Monte Carlo methods provided satisfactory estimation results but faced limitations in complex real-time inference settings such as multi-object tracking with clutter. To address this, we adopt the more efficient variational Bayes as our primary inference paradigm in the subsequent three chapters.

Chapter 4 addresses the issue of large estimation errors in variational Bayes due to the standard mean-field assumption for highly correlated variables. This chapter introduces a novel conditionally factorised variational family featuring an adjustable conditional structure that retains the dependence between desired variables, while

also encompassing the standard mean-field family as a special case. The resulting coordinate ascent updates are derived and approximated using importance sampling. By choosing a finer conditional structure, our algorithm can be guaranteed to achieve a better variational lower bound, offering a flexible trade-off between computational cost and inference accuracy. We discuss and demonstrate this guaranteed performance improvement over the standard mean-field variational Bayes in a simple example.

In the remaining two chapters, we address the challenges of data association for multiple object tracking by employing the non-homogeneous Poisson process (NHPP), a widely used model that elegantly handles objects generating zero or multiple measurements.

Chapter 5 introduces the Variational Bayes Association-based NHPP Tracker (VB-AbNHPP), which efficiently performs tracking, data association, and learning of target and clutter rates with a parallelisable implementation. Additionally, the VB-AbNHPP tracker can be easily extended for online learning of other static parameters, such as object extent, within a general coordinate ascent variational filtering framework developed in this chapter. Detailed derivations of the proposed tracker, including a reasonable initialisation of the variational distribution and the efficient-to-implement (though laborious-to-derive) variational lower bound, are provided. The results verify the effectiveness of the rates learning and demonstrate that the proposed VB-AbNHPP tracker outperforms competing methods in terms of implementation efficiency and tracking accuracy.

Chapter 6 extends the VB-AbNHPP tracker from **Chapter 5** to tackle highly challenging tracking scenarios involving a substantial number of closely-spaced objects and heavy clutter. To address these challenges, this chapter introduces a novel variational localisation strategy that enables quick rediscovery of missed targets within a large surveillance area under extremely heavy clutter. The proposed strategy employs variational Bayes to seek the global optimum in an innovative manner. This strategy is integrated into the standard VB-AbNHPP tracker, along with a proposed novel track loss detection procedure, resulting in a robust VB-AbNHPP tracker that can automatically detect and recover from track loss. In terms of both accuracy and efficiency, this robust VB-AbNHPP tracker significantly outperforms existing trackers in simulated challenging tracking environments.

Ultimately, **Chapter 7** provides a summary of the thesis contributions and outlines potential directions for future research.

1.3 Publication

Parts of this thesis have been submitted or published in the following conference and journal papers:

1. Gan, R., Li, Q. and Godsill, S., 2023. Variational tracking and redetection for closely-spaced objects in heavy clutter. IEEE Transactions on Aerospace and Electronic Systems. Submitted on 24 May 2023.

2. Gan, R.[*], Li, Q.[*] and Godsill, S., 2022, July. A variational Bayes association-based multi-object tracker under the non-homogeneous Poisson measurement process. In 2022 25th International Conference on Information Fusion (FUSION) (pp. 1-8). IEEE.

3. Gan, R. and Godsill, S., 2022, May. Conditionally factorized variational Bayes with importance sampling. ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (pp. 5777-5781). IEEE.

4. Gan, R., Ahmad, B.I. and Godsill, S., 2021. Lévy state-space models for tracking and intent prediction of highly maneuverable objects. IEEE Transactions on Aerospace and Electronic Systems, 57(4).

5. Gan, R. and Godsill, S., 2020, July. α-Stable Lévy state-space models for manoeuvring object tracking. In 2020 IEEE 23rd International Conference on Information Fusion (FUSION), pp. 1-7. IEEE.

6. Gan, R.[*], Liang, J.[*], Ahmad, B.I. and Godsill, S., 2020. Modeling intent and destination prediction within a Bayesian framework: Predictive touch as a usecase. Data-Centric Engineering, 1, pp. e12.

The publications that are relevant but not included in this thesis are listed as follows:

1. Li, Q., Gan, R., and Godsill, S., 2023. A scalable Rao-Blackwellised sequential MCMC Sampler for joint detection and tracking in clutter. 2023 IEEE 26rd International Conference on Information Fusion (FUSION). Accepted.

[*]Gan R. and Li Q. are co-first authors. Gan R. led conceptualization, methodology, implemented the proposed methods and drafted the method and modelling sections. Li Q. led data analysis, implemented all comparison methods, drafted the results and introduction sections. Both reviewed and edited the manuscript.

[*]These authors contributed equally to this work

2. Li, Q., Gan, R., Liang, J. and Godsill, S., 2023. An adaptive and scalable multi-object tracker based on the non-homogeneous Poisson process. IEEE Transactions on Signal Processing.

3. Gan, R., Liang, J., Ahmad, B.I. and Godsill, S., 2019, October. Bayesian intent prediction for fast maneuvering objects using variable rate particle filters. In 2019 IEEE 29th International Workshop on Machine Learning for Signal Processing (MLSP), pp. 1-6. IEEE.

4. Liang, J., Ahmad, B.I., Gan, R., Langdon, P., Hardy, R. and Godsill, S., 2019. On destination prediction based on Markov bridging distributions. IEEE Signal Processing Letters, 26(11), pp. 1663-1667.

Chapter 2

Modelling Intent and Destination Prediction within a Bayesian Framework

In various scenarios, the motion of a tracked object, for example a pointing apparatus, pedestrian, animal, vehicle and others, is driven by achieving a premeditated goal such as reaching a destination. This is albeit the various possible trajectories to this endpoint. This chapter presents a generic Bayesian framework that utilises stochastic models that can capture the influence of intent (namely, destination) on the object behaviour. It leads to simple algorithms to infer, as early as possible, the intended endpoint from noisy sensory observations, with relatively low computational and training data requirements.

In particular, this chapter will discuss several destination-reverting dynamic models and their corresponding intent inference procedures. Part of this chapter was published in [61][1]. The intent inference framework is showcased in the context of novel predictive touch technology for intelligent user interfaces and touchless interactions. The framework can determine, early in an interaction task or pointing gesture, the interface item a user intends to select on a display (e.g. touchscreen), thereby simplifying and speeding up the selection process. This is shown to significantly improve the usability of displays in vehicles, particularly when affected by perturbations from road and driving conditions, and enable intuitive contact-free interactions. Data collected in instrumented vehicles is used to demonstrates the effectiveness of the proposed intent prediction approach.

[1]

2.1 Introduction

In conventional *sensor-level* tracking the objective is typically to estimate the hidden state $\mathbf{x}_t$ of an object of interest (e.g. pointing apparatus, pedestrian, vehicle, vessel, airplane, etc.), where $\mathbf{x}_t$ is the target location, orientation, velocity, higher order kinematics or other spatio-temporal characteristics. This state is assumed to be related to the available noisy sensory measurements (e.g. from camera, radar, inertial measurement units, radio frequency transmissions, global navigation satellite system, acoustic signals, etc.) as per a defined observation model. A plethora of well-established algorithms for estimating $\mathbf{x}_t$ exist, including from multiple data sources, see [62, 4, 5]. They often implicitly assume that the object moves in an unpremeditated manner and suitable motion models are accordingly employed.

In this chapter, the main objective is not to estimate the state $\mathbf{x}_t$, but instead to infer the underlying intent that is driving the object motion, namely its destination. This capitalises on the premise that the target motion (e.g. the trajectory followed by a pointing finger whilst interacting with a display) is dictated by the intended endpoint (e.g. the sought interface item), and that the destination influence on the target movements can be modelled. Therefore, the sought probabilistic modelling and destination predictor(s) belong to a higher system level compared with the sensor-level tracking techniques, hence dubbed *meta-level tracking* algorithms [6]. They have several applications, such as in surveillance, human computer interaction, robotics and others, since such meta-level approaches can facilitate automated decision making, resources allocation and informed future action planning. They offer a more integrated viewpoint of a scene where intents can be automatically learnt and conflict or opportunities can be identified in a timely manner. The HCI technology, dubbed predictive touch, is used here as an application or motivation for the proposed Bayesian meta-level inference framework. Nonetheless, this approach can be applied in numerous other areas and scenarios.

2.1.1 Predictive touch

As a motivating application of the intent inference framework presented in this chapter, we briefly introduce the predictive touch technology. For more information about predictive touch, please refer to [61, 7].

Predictive touch is an emerging HCI technology for intelligent displays and touchless interactions that predicts the intended interface component selection, notably early in the pointing-selection task, based on available freehand pointing movements in 3D

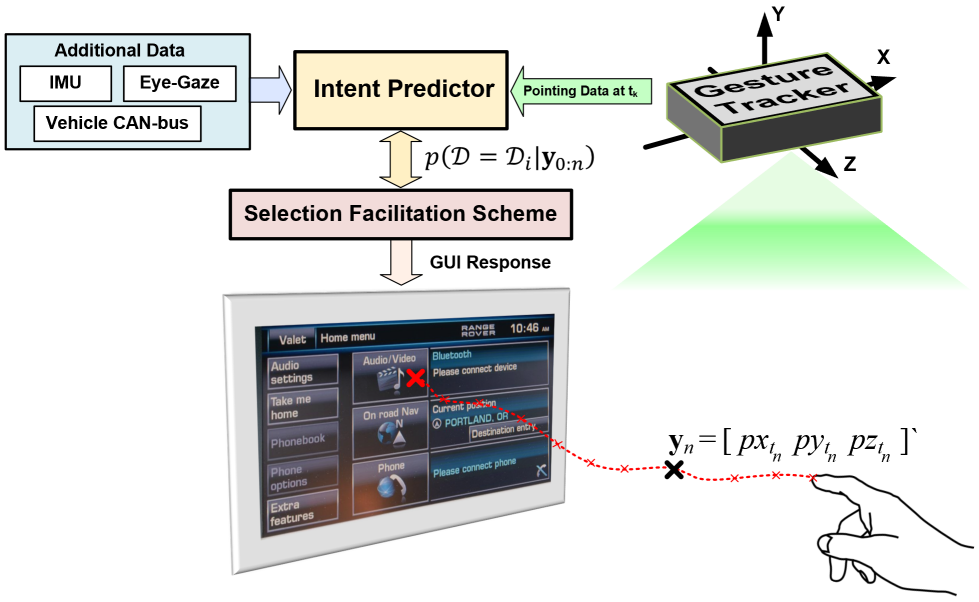


Fig. 2.1 Block diagram of an in-vehicle predictive touch system. The dotted line is a recorded full in-car pointing trajectory. The gesture tracker (sensor is facing downwards to increase the region of coverage and minimise occlusions) provides at time t_n the pointing finger/hand Cartesian coordinates along the x, y and z axes, denoted by $\mathbf{y}_n$.

and other sensory data, such as eye-gaze [7]. This technology can significantly reduce effort and distractions associated with using in-vehicle displays while driving [63] and improve usability of displays in moving platforms.

As illustrated in Fig. 2.1, predictive touch systems typically comprise four main modules:

- *Pointing Gesture Tracker*: Provides the pointing hand/finger(s) location in 3D in real-time. Gesture trackers are increasingly common in automotive, gaming, and infotainment applications due to advancements in sensing and computer-vision systems. The data used in this thesis are collected by the Leap Motion controller, which employs optical sensors and infrared light to monitor an approximately hemispherical area with an effective tracking range of 60cm. This sensor data is subsequently integrated with an internal model of the hand and fingers to output the position of a fingertip (the $\mathbf{y}_n$ in Fig. 2.1). In instances where a finger or its segments are obscured, the system estimates the finger's characteristics based on recent observations and the anatomical hand model. It's worth noting that the gesture tracker may also provide information on the direction of finger pointing, the positions of other fingertips, hand orientation, and arm movements [64]. Although this additional information could be valuable for the intent inference

task under consideration, we opt not to utilise it here to maintain both simplicity and a level of generalisability to broader tracking contexts such as vehicles and animals.

- *Intent Predictor*: Calculates the likelihood of each selectable interface icon being the intended destination using available pointing trajectory data.
- *Selection Facilitation*: Simplifies and expedites the selection task based on prediction results through various facilitation schemes. Studies have shown that pointing time can be reduced by over 30% and effort/workload halved with predictive touch [7].
- *Additional Data*: Utilises available sensory data to improve prediction results, such as inertial measurements, eye-gaze data, and environmental data.

Predictive touch allows users to interact with displays without physically touching them, improving usability and performance. The technology also enables interaction with new display technologies, such as head-up displays, holograms, and 3D projections. This novel HCI solution uses the intuitive free hand pointing gestures and intrinsically relies on predicting the user intent, rather than using the pointing finger/arm location or orientation as a pointing apparatus as in [65]. Unlike gesture-recognition-based interactions [66], predictive touch does not require users to pre-learn specific gestures and offers design flexibility in display placement and GUI design. This promotes inclusive design practices by tailoring display operation to user requirements through configurable prediction algorithms and facilitation schemes.

2.1.2 Related work and contributions

The Bayesian framework for intent prediction presented in this chapter was introduced in [67] and [68] for predictive touch and other applications. It treats the problem within an object tracking formulation, albeit not necessarily seeking state estimation, such that the influence of intended destination is captured by utilising suitable stochastic motion model with a few unknown parameters. The latter parameters can be estimated from a small number of example motion patterns or trajectories. Linear Gaussian systems were considered in the aforementioned papers and more recently nonlinear behaviour due to external forces (e.g. jumps and jolts in the pointing movements due to the road/driving conditions) was briefly addressed in [69].

The main objective of this chapter is to explain the rationale of Bayesian intent inference framework and provide a unified treatment of the intent prediction task. The contributions include:

1. An overview of linear and nonlinear (albeit within a conditionally linear formulation) motion models and systems, along with their corresponding intent inference procedures, is provided. Notably, the developments of the Equilibrium Reverting Acceleration model (Section 2.3.1.1), the mean-reverting jump diffusion model (Section 2.3.2), and the intent inference procedure that utilises particle filter-based likelihood computation (Section 2.4.2), are all original contributions of mine, as detailed in [69];

2. Among the intent-driven models discussed, the pseudo-observation *bridging distribution* (BD) is articulated as a distinct stochastic process. This is in contrast to its original presentation in [70], where it was entangled with the intent inference procedure. The transition density of this stochastic process is explicitly given in equations (2.15) and (2.18) for the first time. Following this, an intent inference procedure that is based on evaluating the likelihood of this process is introduced in Algorithm 1;

3. The newly tested bridged (nearly) constant acceleration (CA) dynamic model, which has not been incorporated in the new or original BD formulations [70, 68] before, is evaluated here for various BD formulations and shown to deliver the highest prediction performance for a predictive touch system;

4. We benchmark various prediction models using significantly larger data set of pointing gestures recorded in instrumented vehicles under various road-driving conditions.

Concerning the second contribution, it's pivotal to emphasise that the motivation behind reformulating the BD model (in Section 2.3.1.2) and introducing its inference procedure Algorithm 1 isn't solely about outclassing other BD variants [68, 70] in either accuracy or efficiency. Notably, all BD methodologies should exhibit comparable inference accuracy given their analogous foundational principles. Our primary aims remain: 1) To provide a stochastic process interpretation for the pseudo-observation BD model, 2) To deliver more intuitive inference strategies rooted solely in evaluating the likelihoods of pseudo-observation BD processes that are driven by varying destinations.

In the tracking area, incorporating *known* predictive information to improve the accuracy of state estimation has a long history, e.g. [71] and [72]. Additionally, mean-

reverting models such as those derived from an Ornstein–Uhlenbeck (OU) process (defined later in Section 2.3.1.1 to satisfy the stochastic differential equation (2.4)), with known means, were to better estimate behavior of certain objects, e.g. vessel [73] or financial time series data in [74]. Also, the use of stochastic context-free grammar (SCFG) and conditionally Markov process/reciprocal process has been proposed to predict intent as in [6, 75, 76]. In this chapter, the destination (i.e. intent) is assumed to be unknown and predictors are developed to infer it. The adopted formulation here leads to significantly simpler algorithms with no constraints on the trajectory followed by the object (e.g. freehand pointing finger), unlike those using SCGF which discretise the state space. The employed continuous state space models within the introduced Bayesian framework, such as OU process-based model and bridging distributions (both are detailed in Section 2.3), enable treating asynchronous sensory measurements. A noteworthy fact is that the bridging distribution can be viewed as special case of conditionally Markov models in [76] under certain assumptions.

On the other hand, modelling and inferring complex intentions, such as drivers behaviours at junctions, pedestrians at crosswalks and human daily activities, can be tackled with data-driven or classification approaches, possibly combined with *a priori* learnt pattern of life. They assume the availability of sufficiently complete and diverse training data sets with several well-established such prediction techniques, for example [77], [78] and [79]. However, in this chapter the objective is to develop a simple and computationally efficient destination prediction algorithm where limited training data is available. For example, it can be very challenging and expensive to collect data sets of 3D freehand pointing gestures that sufficiently sample possible paths/trajectories to the display, starting locations of the gesture (e.g. steering wheel, armrest and others), road/driving conditions, context of use, user interface design, screen size/reach, etc. Instead, suitable state space models are employed here, albeit with a few unknown parameters, as is common in object tracking. They enable modelling and robustly inferring the intended endpoint of a tracked object, especially that the possible intentions are a finite set of nominal destinations, e.g. selectable interface items. Subsequently, the introduced Bayesian intent predictors have minimal training requirements.

2.1.3 Layout

The remainder of this chapter is organised as follows. Section 2.2 presents the fundamental assumptions and overall inference framework. Various approaches to modelling intent and the dynamic model are described in Section 2.3. Destination predictors

for linear and nonlinear settings are then outlined in Section 2.4. Results using real pointing data, recorded by in-vehicle predictive touch prototypes under various road conditions, are presented in Section 2.5 and conclusions are drawn in Section 2.6.

2.2 Fundamental assumptions and Bayesian intent inference framework

In this chapter, the destination inference problem is addressed within a Bayesian framework, focusing specifically on scenarios with a finite number of possible destinations to be inferred. The inference framework that accommodates an infinite number of possible destinations and/or allows a time-varying intent will be presented in the next chapter. This section sets up fundamental assumptions for the considered destination inference problem and introduces the Bayesian intent inference framework.

2.2.1 Notations and fundamental assumptions

The considered meta-level tracking problem adopts some of the same basic assumptions as those in a typical sensor-level tracking problem. Specifically,, we assume that the tracked object (e.g. pointing finger) moves in continuous-time, and the sensor (e.g. gesture tracker) observes it at discrete time instants $t_1 < t_2 < t_3 < \cdots < t_N$. The sensory measurement received at time instant t_n is denoted by the column vector $\mathbf{y}_n$. We denote the column vector $\mathbf{x}_t$ as the tracked target's kinematics state at an arbitrary time instant t, and abbreviate $\mathbf{x}_{t_n}$, i.e. the kinematics state captured by the measurement $\mathbf{y}_n$, as $\mathbf{x}_n$. We define the abbreviation $\mathbf{y}_{1:n} = [\mathbf{y}_1^\top, \mathbf{y}_2^\top, ..., \mathbf{y}_n^\top]^\top$, and similarly, $\mathbf{x}_{1:n} = [\mathbf{x}_1^\top, \mathbf{x}_2^\top, ..., \mathbf{x}_n^\top]^\top$.

2.2.2 Bayesian intent inference framework

Consider a scenario with $N_{\mathcal{D}}$ nominal endpoints, indexed by $i = 1, 2, ..., N_D$, where one of these endpoints represents the true destination of the tracked target. In this thesis, the endpoint/destination may be a fixed point, or a distribution of possible points, defined perhaps by a Gaussian density, as will be made clear in the following contexts. Denote the index of the intended destination of the tracked target as a discrete random variable $\mathcal{D}$, and evidently, its domain is $\mathcal{D} \in \{1, 2, ..., N_D\}$. This implies that the event of the i-th nominal endpoint being the intended destination can be described by $\mathcal{D} = i$.

The objective here is to infer the tracked target's intended destination (represented by its index $\mathcal{D}$) using noisy sensory measurements observed from the target. To this end, it is necessary to establish a probabilistic relationship between these measurements and the target's intended destination $\mathcal{D}$. A reasonable approach to develop this relationship is by postulating that the intended destination $\mathcal{D}$ influences the target's kinematic state transitions, acting as an underlying parameter. This is because the intended destination would naturally impact the target's movement. Consequently, through the observation model, the intended destination $\mathcal{D}$ also affects the sensory measurements, enabling the inference process. This approach transforms the destination inference problem into a parameter learning/model calibration problem, which has been extensively studied in the field of signal processing.

Subsequently, we assume that the random variable $\mathcal{D}$ is one of the dynamic model's parameters, conditioned on which, the dynamic model reflects the influence of the intended destination on the target's movement, e.g. having a tendency to move to the nominal endpoint indexed by $\mathcal{D}$. In particular, our knowledge of the basic information on the $N_{\mathcal{D}}$ potential destinations, such as location and geometry, can be used to construct/define such destination-driven dynamic models. This will be detailed in Section 2.3.

Now, inferring the intended destination from the $N_{\mathcal{D}}$ nominal endpoints becomes inferring the underlying model parameter $\mathcal{D}$ from its domain $\{1, 2, ..., N_D\}$. Subsequently, the objective of our Bayesian intent inference is to *sequentially* calculate the probability of each endpoint (e.g. selectable interface components) being the intended destination, i.e. $\mathcal{D} = i$, $i = 1, 2, ..., N_D$, at the current/latest time instant t_n, thus $p(\mathcal{D} = i \mid \mathbf{y}_{1:n})$, $\quad i = 1, 2, ..., N_D$, from the available sensory measurements $\mathbf{y}_{1:n}$. We recall that in a predictive touch system observations $\mathbf{y}_{1:n}$ are provided by the gesture tracker and other sensors at the successive time instants $t_1, ..., t_n$, for instance $\mathbf{y}_n$ is the 3-D Cartesian coordinates of the pointing finger/hand at t_n as in Fig. 2.1. For each $i = 1, 2, ..., N_D$ and per Bayes' rule, we have

$$p(\mathcal{D} = i|\mathbf{y}_{1:n}) \propto p(\mathbf{y}_{1:n}|\mathcal{D} = i)p(\mathcal{D} = i), \tag{2.1}$$

where $p(\mathcal{D} = i)$ is the prior probability of the i-th endpoint being the intended destination. In predictive touch this prior can be attained from semantic data, frequency of use, interface design, other sensory data, etc. The task of the inference module (i.e. Intent Predictor in Fig. 2.1) at t_n is hence to estimate the likelihoods $p(\mathbf{y}_{1:n}|\mathcal{D} = i)$, $i = 1, 2, ..., N_{\mathcal{D}}$. This makes the Bayesian formulation particularly appealing since additional contextual information can be easily incorporated, whenever available.

2.2.3 Observation model

We assume the available sensory measurement $\mathbf{y}_n$ (e.g. gesture-tracker output) is a noisy observation of the true hidden state $\mathbf{x}_n$ (e.g. pointing finger actual location). Here and for simplicity, a linear and Gaussian measurement model is assumed. In a state space form, it is described at time t_n by

$$\mathbf{y}_n = \mathbf{H}\mathbf{x}_n + \mathbf{w}_n, \tag{2.2}$$

where $\mathbf{w}_n$ is the zero mean i.i.d. Gaussian noise where $\mathbf{w}_n \sim \mathcal{N}(\mathbf{0}, \mathbf{V}_n)$. For instance, if the gesture tracker provides locations of the pointing finger in 3-D and latent state $\mathbf{x}_n \in \mathbb{R}^3$ consists only of the object location, the observation matrix in (2.2) would be a 3×3 identity matrix, denoted as $\mathbf{H} = \mathbf{I}_3$. The noise covariance matrix $\mathbf{V}_n$ is specified by the tracker accuracy, i.e. in terms of determining the pointing finger position.

Note that the measurement model in (2.2) is formulated with $\mathbf{x}_n$ but without information about the intended destination. This is consistent with the fact that most sensors can directly observe information such as the kinematic state, but not the underlying intention. Subsequently, the assumed measurement model in (2.2) implies the following conditionally independent factorisation:

$$p(\mathbf{y}_{1:n}|\mathbf{x}_{1:n}, \mathcal{D} = i) = \prod_{k=1}^{n} p(\mathbf{y}_k|\mathbf{x}_k). \tag{2.3}$$

2.3 Destination-driven motion models

Recall that, based on our assumption in Section 2.2.2, the dynamic model of the tracked target is conditioned on the intended destination $\mathcal{D}$, which serves as a parameter of the model and drives the target's movement. Then, a key challenge within the introduced inference framework is to employ suitable motion models that represent the effect of intent on the object's motion. In other words, we need to define an appropriate transition density for the destination-driven dynamic model $p(\mathbf{x}_n|\mathbf{x}_{1:n-1}, \mathcal{D})$ for all possible intended destinations $\mathcal{D} \in \{1, 2, ..., N_D\}$. The object motion in our considered scenario demonstrates a tendency to move towards the destination, such as the fingertip moving towards an intended item on a display. Consequently, the dynamic model governed by the transition $p(\mathbf{x}_n|\mathbf{x}_{1:n-1}, \mathcal{D} = i)$ should exhibit a mean-reverting behavior towards the i-th nominal endpoint. In this section, we investigate and present various options for constructing such models.

In this chapter, the destination-driven motion model is formulated as continuous-time Markov or conditional Markov process. This is because it aligns with the continuous nature of the object's movement being tracked; and the (conditional) Markov property is commonly assumed for computational convenience in sequential inference tasks. As a result, the transition probability $p(\mathbf{x}_n|\mathbf{x}_{1:n-1}, \mathcal{D} = i)$ simplifies to $p(\mathbf{x}_n|\mathbf{x}_{n-1}, \mathcal{D} = i)$ for all $i = 1, 2, \ldots, N_D$ in the case of a Markov model. For the conditional Markov model, the transition probability $p(\mathbf{x}_n|\mathbf{x}_{1:n-1}, \mathcal{D} = i, \mathcal{C})$ reduces to $p(\mathbf{x}_n|\mathbf{x}_{n-1}, \mathcal{D} = i, \mathcal{C})$, where $\mathcal{C}$ is a specific condition that will be elaborated upon in Sections 2.3.1.2 and 2.3.2.

Moreover, the transition $p(\mathbf{x}_n|\mathbf{x}_{1:n-1}, \mathcal{D} = i)$ is inherently stochastic, which is necessary for carrying out probabilistic inference with the received measurements. The stochastic nature of the transition also reflects the fact that object motion, such as the movement of a fingertip towards an intended item on a display, is driven by a very complex sensorimotor system, capable of autonomous action based on various modalities (e.g. vision and can utilise feedback on the action) and is also subjected to various constraints (e.g. to optimise action required to deliver/predict smooth movement trajectories and minimise the variance of the eye or arm's position, in the presence of biological noise due to mechanical properties of muscles) and possibly perturbed by external forces such as due to road/driving conditions or walking, see [80].

2.3.1 Linear Gaussian motion models

Simple approximate motion models, which focus on inferring intent rather than providing a highly accurate representation of the object motion, can suffice for the task of destination prediction. Under this assertion, linear Gaussian models can be particularly favourable since they can be easily formulated and lead to computationally efficient prediction algorithms, compared with non-linear non-Gaussian models [81, 62]. Next, two classes of linear Gaussian intent-driven models, namely OU process-based model and (arrival time conditioned) bridging distributions, are introduced.

2.3.1.1 Ornstein–Uhlenbeck process-based models

The OU process with mean reverting property offers an effective way to model the destination-driven behaviour. By setting the mean term of the underlying model according to the destination information, the target would revert to the premeditated endpoint and finally arrive somewhere nearby. We now discuss the formulation for the

model that reverts to the i-th endpoint, i.e. conditioned on the event $\mathcal{D} = i$. Under this setting, the OU-based models can be described in continuous time by the following linear time invariant stochastic differential equation (SDE),

$$d\mathbf{x}_t = \mathbf{A}(\boldsymbol{\mu}_i - \mathbf{x}_t)dt + \boldsymbol{\sigma} d\boldsymbol{\beta}_t, \tag{2.4}$$

where $\boldsymbol{\beta}_t$ is a multivariate standard Wiener process. Before introducing the parameterisations for 3-D pointing movements, an illustration is provided below to clarify the intuition behind this SDE.

Breaking down the Ornstein–Uhlenbeck SDE (2.4) **with simple examples**

In SDE (2.4), setting $\boldsymbol{\sigma}$ to $\mathbf{0}$ transforms the equation into the ordinary differential equation (ODE):

$$\frac{d\mathbf{x}_t}{dt} = \mathbf{A}(\boldsymbol{\mu}_i - \mathbf{x}_t). \tag{2.5}$$

This ODE can depict several dynamic systems with physical implications when we appropriately define $\mathbf{x}_t$ and the parameters $\mathbf{A}, \boldsymbol{\mu}_i$. As an illustration, when

$$\mathbf{x}_t = \begin{bmatrix} x_t \\ \dot{x}_t \end{bmatrix}, \quad \mathbf{A} = \begin{bmatrix} 0 & -1 \\ \frac{\eta}{m} & \frac{\rho}{m} \end{bmatrix}, \quad \boldsymbol{\mu}_i = \begin{bmatrix} l \\ 0 \end{bmatrix}, \tag{2.6}$$

our ODE (2.5) then reads:

$$\begin{aligned} \frac{dx_t}{dt} &= \dot{x}_t, \\ m\frac{d\dot{x}_t}{dt} &= \eta(l - x_t) - \rho\dot{x}_t. \end{aligned}$$

Given that η, ρ, and m are all positive, this captures the dynamics of a point object of mass m moving in one-dimensional space. Here, x_t represents the position and $\dot{x}_t$ denotes the velocity. The object is drawn towards position l by a force $\eta(l - x_t)$ (akin to the spring's restoring force). Concurrently, a force $\rho\dot{x}_t$ opposes its motion (analogous to air resistance or damping). To introduce stochasticity into this system, one might imagine to add an independent, extremely small Gaussian white noise into the applied force for every infinitesimal time increment. Formally, this stochastic system can be represented by the SDE (2.4). Use the same $\mathbf{x}_t, \mathbf{A}$ and $\boldsymbol{\mu}_i$ as in (2.6), but now with $\boldsymbol{\sigma} = [0, \sigma]^\top$, and $\boldsymbol{\beta}_t$ as a scalar standard Wiener process.

For a 3-D pointing movement, we have the following parameterisations for the SDE (2.4): $\mathbf{x}_t = [\mathbf{x}_{t,1}^\top, \mathbf{x}_{t,2}^\top, \mathbf{x}_{t,3}^\top]^\top$, with $\mathbf{x}_{t,s} \in \mathbb{R}^2$ (position and velocity) or $\mathbb{R}^3$ (position, velocity and acceleration), $s = 1, 2, 3$, and

$$\mathbf{A} = \text{diag}\{\mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3\}, \quad \boldsymbol{\mu}_i = [\boldsymbol{\mu}_{i,1}, \boldsymbol{\mu}_{i,2}, \boldsymbol{\mu}_{i,3}]^\top, \quad \boldsymbol{\sigma} = \begin{bmatrix} \boldsymbol{\sigma}_1 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\sigma}_2 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \boldsymbol{\sigma}_3 \end{bmatrix}, \quad \boldsymbol{\beta}_t = \begin{bmatrix} \beta_{t,1} \\ \beta_{t,2} \\ \beta_{t,3} \end{bmatrix}. \tag{2.7}$$

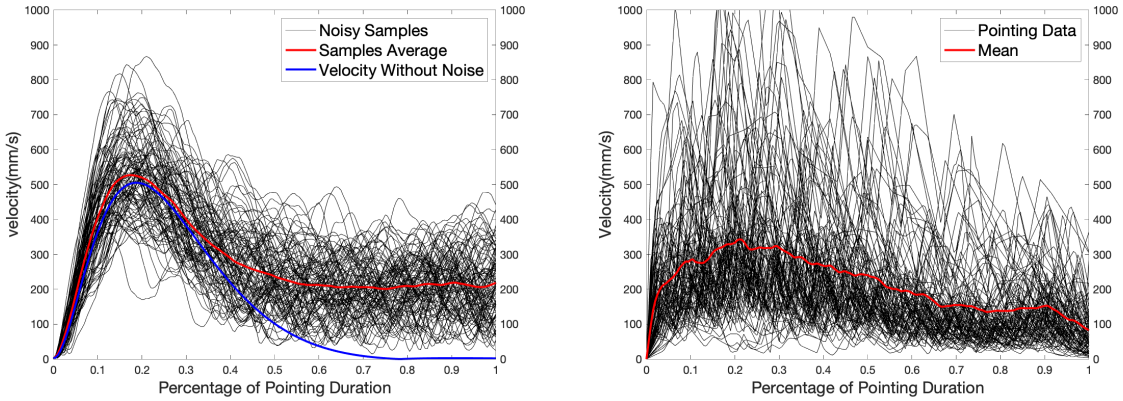
Different kinematics included in each 'sub-state' $\mathbf{x}_{t,s}$ along with the corresponding parameters lead to distinct SDEs as per equation (2.4), for instance: a) the Mean Reverting Diffusion (MRD) model which only includes position in the state [67], b) Equilibrium Reverting Velocity (ERV) that model position and velocity [67], and c) Equilibrium Reverting Acceleration (ERA) representing position, velocity and acceleration [69]. These three models have similar mean reverting behaviour, that is, the state will revert to the mean term $\boldsymbol{\mu}_i$, e.g. set as the destination position for MRD and with (nearly) zero velocity and acceleration for ERV and ERA, respectively. Here we only discuss the setup for ERA model for simplicity, while other models follow the similar rationale, refer to [67] for further details. For ERA, the sub-matrices and vectors in equation (2.7) for the s^{th} dimension are

$$\mathbf{A}_s = \begin{bmatrix} 0 & -1 & 0 \\ 0 & 0 & -1 \\ \eta & \rho & \gamma \end{bmatrix}, \quad \boldsymbol{\mu}_{i,s} = [p_{i,s}, 0, 0], \quad \mathbf{x}_{t,s} = \begin{bmatrix} x_{t,s} \\ \dot{x}_{t,s} \\ \ddot{x}_{t,s} \end{bmatrix}, \quad \boldsymbol{\sigma}_s = \begin{bmatrix} 0 \\ 0 \\ \sigma \end{bmatrix}, \tag{2.8}$$

where $p_{i,s}$ is the position of the i-th nominal endpoint in the s-th dimension, and $\dot{x}_{t,s}, \ddot{x}_{t,s}$ denote the second and third derivative (velocity and acceleration) of $x_{t,s}$. The above setup assumes independent transitions for each coordinate, specifically, it can be specified by the following SDE,

$$d\ddot{x}_{t,s} = \eta(p_{i,s} - x_{t,s})dt - \rho\dot{x}_{t,s}dt - \gamma\ddot{x}_{t,s}dt + \sigma d\beta_{t,s}. \tag{2.9}$$

One can see that the object motion governed by such an SDE will initially gravitate to the destination position (i.e. $p_{i,s}$ prescribed in the mean vector $\boldsymbol{\mu}_{i,s}$ of this OU process) with increasing acceleration due to the positive reversion factor η, then the positive damping factor ρ and γ would guarantee the target slows down and arrives the destination in an equilibrium state, with nearly zero velocity and acceleration. To ensure wide-sense stationarity of the system in (2.9), the model parameters should have



(a) ERA velocity samples (b) Real pointing data

Fig. 2.2 The 3D norm velocity profile generated by the ERA model is shown in (a), where the black lines are 100 random realisations, the red line is the mean of them, and the blue line shows the deterministic transition of the norm velocity of the same ERA model. (b) shows the velocity profile from 95 real pointing data, where the red line is the mean trajectory.

the following relationship $0 < \eta < \rho\gamma$. This can be proved by applying *Routh-Hurwitz* stability criterion [82] for a third-order system.

The aforementioned velocity behaviour can be demonstrated as the blue line in Fig. 2.2(a), which is the deterministic transition (i.e. with σ in (2.9) being zero) of the norm velocity of the ERA model. The norm velocities of an ERA model depicted in Fig. 2.2(a), i.e. sample realisations as well as their mean, are generated from the parameters manually tuned to maximise the intent prediction accuracy. They noticeably capture, on average, an overall profile similar to that exhibited by the real pointing gesture data shown in 2.2(b).

Solving (2.4) yields the general discrete linear Gaussian Markov transition function for all three models (MRD, ERV and ERA) as per,

$$p(\mathbf{x}_n|\mathbf{x}_{n-1}, \mathcal{D} = i) = \mathcal{N}(\mathbf{x}_n|\mathbf{F}_{i,h}\mathbf{x}_{n-1} + \mathbf{M}_{i,h}, \mathbf{Q}_{i,h}) \tag{2.10}$$

such that

$$\begin{aligned} h &= t_n - t_{n-1}, \\ \mathbf{F}_{i,h} &= e^{-\mathbf{A}h}, \\ \mathbf{M}_{i,h} &= (\mathbf{I} - e^{-\mathbf{A}h})\boldsymbol{\mu}_i, \\ \mathbf{Q}_{i,h} &= \int_0^h e^{-\mathbf{A}(h-v)}\boldsymbol{\sigma}\boldsymbol{\sigma}^\top e^{-\mathbf{A}^\top(h-v)}dv. \end{aligned} \tag{2.11}$$

Whereas, $\mathbf{A}$, $\boldsymbol{\mu}_i$, and $\boldsymbol{\sigma}$ are parameters set for the specific model, $\mathbf{I}$ is the identity matrix with the corresponding size. The derivation of this solution and calculation for $\mathbf{Q}_{i,h}$ can be found in [67] and references therein.

A notable extension of the OU-based model introduced above involves incorporating the geometric information of the destination. This can be achieved by modelling the endpoint position $p_{i,s}$, $s = 1, 2, 3$, in (2.8) as a static random variable, where its covariance accounts for the destination's geometry. Specifically, when this variable is Gaussian, the tractability of the inference task can be maintained. This extension also enables direct inference on the intended destination's position without restricting it to one of the finite nominal endpoints. An example of such a destination prediction method will be introduced in the next chapter.

2.3.1.2 Bridging distributions

While the destination information is modelled above by the mean of the OU process, another approach to incorporate such knowledge can be provided by the bridging distributions method. This is particularly relevant if we use a known or legacy motion model, which does not encapsulate the influence of intent on the object motion, as seen in numerous models found in tracking literature, such as the nearly constant velocity (CV) and acceleration (CA) models; see [10] for a comprehensive overview. Additionally, in some scenarios an OU process might not accurately characterise the destination reverting behaviour of the tracked object. In such cases, bridging distributions permit more free underlying motion dynamics and at the same time ensures the object arrival at/near its endpoint.

Bridging distributions (BDs), introduced as a terminology in [68], and later further developed in [70], represent a destination-driven model/stochastic process. The BD definitions from both [68] and [70] have slight variations, which will be detailed in the following contexts. Generally, BDs capture the destination influence on the target behaviour by constructing a Markov bridge between the intended endpoint and the target current state at t_n. This capitalises on the premise that the trajectory

followed by the object (e.g. pointing finger) must terminate at the endpoint (on-display selectable interface item), at arrival time $\mathcal{T}$, despite the random behaviour between the current time step t_n and $\mathcal{T}$. BDs accordingly introduce this knowledge into a motion model via a prior and facilitates destination-aware behaviour modelling without requiring the development of specialised stochastic processes that are intrinsically intent-driven. Nonetheless, BDs may be applied to OU-type models for means dictated by a destination or not, for endpoint-driven OU process BDs can reduce their sensitivity to parameterisation as discussed in [68].

Assuming that the target will reach the destination at time $t_N = \mathcal{T}$, a terminal state is defined as $\mathbf{x}_N$. At their core, BDs represent a stochastic process formed by altering a standard Markov process, based on available destination knowledge. One significant effect of this alteration, or transformation, is a heightened certainty about the prior of terminal state $\mathbf{x}_N$. The state transition density of BDs, which conditions on the destination and the arrival time, can be expressed as the conditional distribution $p(\mathbf{x}_n|\mathbf{x}_{n-1}, \mathcal{D} = i, \mathcal{T})$. There exists several ways of finding this conditional density and they may differ based on the made assumption(s). For example, [68] assumes the terminal state $\mathbf{x}_N$ has exactly the same position as the i-th nominal endpoint, and the destination-related information is introduced via a Gaussian prior at t_0, $p(\mathbf{x}_N|\mathcal{D} = i, \mathcal{T}) = \mathcal{N}(\mathbf{x}_N|\mathbf{a}_i, \mathbf{\Sigma}_i)$ with $\mathbf{a}_i$ being the mean, $\mathbf{\Sigma}_i$ the covariance matrix and $i = 1, 2, \ldots, N_{\mathcal{D}}$. This covariance can model the size and orientation of the endpoint, allowing destinations in the BD model to be represented as regions rather than single spatial points. Incorporating this uncertainty in the destination is particularly important for BD destination-driven models. Without such uncertainty, the kinematic state at time instant $\mathcal{T}$ would be a deterministic point that cannot be corrected by any noisy measurements. Additionally, the model assumes that the desired transition density, when conditioned on the terminal state (i.e. $p(\mathbf{x}_n|\mathbf{x}_{n-1}, \mathcal{D} = i, \mathcal{T}, \mathbf{x}_N)$), is equivalent to the standard conditional Markov transition density $p(\mathbf{x}_n|\mathbf{x}_{n-1}, \mathbf{x}_N, \mathcal{T})$, which can be evaluated as follows,

$$p(\mathbf{x}_n|\mathbf{x}_{n-1}, \mathbf{x}_N, \mathcal{T}) \propto p(\mathbf{x}_n|\mathbf{x}_{n-1})p(\mathbf{x}_N|\mathbf{x}_n, \mathcal{T}), \tag{2.12}$$

where the Markov assumption $p(\mathbf{x}_N|\mathbf{x}_n, \mathbf{x}_{n-1}, \mathcal{T}) = p(\mathbf{x}_N|\mathbf{x}_n, \mathcal{T})$ is used. Given the fact that the terminal state $\mathbf{x}_N$ is fixed, one can construct a joint state vector $\mathbf{z}_n = [\mathbf{x}_n, \mathbf{x}_N]^\top$ and obtain the transition density for $\mathbf{z}_n$ accordingly. The joint state transition will ultimately lead $\mathbf{x}_n$ to its terminal state $\mathbf{x}_N$ which follows the prior $p(\mathbf{x}_N|\mathcal{D} = i, \mathcal{T})$. When observations are available, such a construction of $\mathbf{z}_n$ permits a joint estimation on destination and kinematic state.

An alternative formulation of BDs can be found in [70], in which the destination information is interpreted as a 'pseudo-observation' instead of as a state prior. Specifically, a linear and Gaussian pseudo-observation model,

$$p(\tilde{\mathbf{y}}_N^i = \mathbf{a}_i|\mathbf{x}_N) = \mathcal{N}(\mathbf{a}_i|\tilde{\mathbf{G}}\mathbf{x}_N, \mathbf{\Sigma}_i), \tag{2.13}$$

was considered, where $\tilde{\mathbf{y}}_N^i$ is a pseudo-observation of the terminal state $\mathbf{x}_N$. The pseudo-observation matrix, $\tilde{\mathbf{G}}$, relates to the available information about the terminal state $\mathbf{x}_N$. Our understanding of the distribution of $\tilde{\mathbf{G}}\mathbf{x}_N$ is represented by $\mathcal{N}(\mathbf{a}_i, \mathbf{\Sigma}_i)$. As an illustration, when we have knowledge only of the destination's spatial aspects, $\tilde{\mathbf{G}}$ should extract the positional details from $\mathbf{x}_N$. Meanwhile, $\mathbf{a}_i$ and $\mathbf{\Sigma}_i$ should respectively denote the mean and covariance of the i-th endpoint's region. Subsequently, the sought conditional Markov transition density is defined as

$$p(\mathbf{x}_n|\mathbf{x}_{n-1}, \mathcal{D} = i, \mathcal{T}) = p(\mathbf{x}_n|\mathbf{x}_{n-1}, \tilde{\mathbf{y}}_N^i = \mathbf{a}_i, \mathcal{T}). \tag{2.14}$$

This transition is implicitly used in [70], Algorithm 2 to predict the destination. Here we explicitly derive its detailed form:

$$\begin{aligned} p(\mathbf{x}_n|\mathbf{x}_{n-1}, \mathcal{D} = i, \mathcal{T}) &= p(\mathbf{x}_n|\mathbf{x}_{n-1}, \tilde{\mathbf{y}}_N^i = \mathbf{a}_i, \mathcal{T}) \\ &\propto p(\mathbf{x}_n, \tilde{\mathbf{y}}_N^i = \mathbf{a}_i|\mathbf{x}_{n-1}, \mathcal{T}) \\ &= \int p(\mathbf{x}_n, \mathbf{x}_N, \tilde{\mathbf{y}}_N^i = \mathbf{a}_i|\mathbf{x}_{n-1}, \mathcal{T}) d\mathbf{x}_N \\ &= \int p(\tilde{\mathbf{y}}_N^i = \mathbf{a}_i|\mathbf{x}_N) p(\mathbf{x}_N|\mathbf{x}_n, \mathcal{T}) p(\mathbf{x}_n|\mathbf{x}_{n-1}) d\mathbf{x}_N. \end{aligned} \tag{2.15}$$

Similar to (2.12), the Markov assumption $p(\mathbf{x}_N|\mathbf{x}_n, \mathbf{x}_{n-1}, \mathcal{T}) = p(\mathbf{x}_N|\mathbf{x}_n, \mathcal{T})$ is applied here. Moreover, using techniques such as the d-separation principle [12], it can be demonstrated that the transition on the right-hand side of (2.14) is conditional Markovian. Specifically, $p(\mathbf{x}_n|\mathbf{x}_{n-1}, \mathbf{x}_{1:n-2}, \tilde{\mathbf{y}}_N^i = \mathbf{a}_i, \mathcal{T}) = p(\mathbf{x}_n|\mathbf{x}_{n-1}, \tilde{\mathbf{y}}_N^i = \mathbf{a}_i, \mathcal{T})$. As a result, the sought transition density $p(\mathbf{x}_n|\mathbf{x}_{n-1}, \mathcal{D} = i, \mathcal{T})$ defined by (2.14) also exhibits the conditional Markov property. This ensures that, when conditioned on $\mathcal{T}$, the exact filtering procedure with this stochastic process can be efficiently executed by the recursive Bayesian filter, which will be employed for intent inference in Section 2.4.1.2.

Motivated by the pseudo-observation-based formulation of BDs, in this chapter we introduce a new intent prediction algorithm which utilises (2.15) as its main ingredient. Similar to [70] and [68], we will focus on linear Gaussian models because they lead to analytically tractable results. First, consider the following linear time invariant SDE,

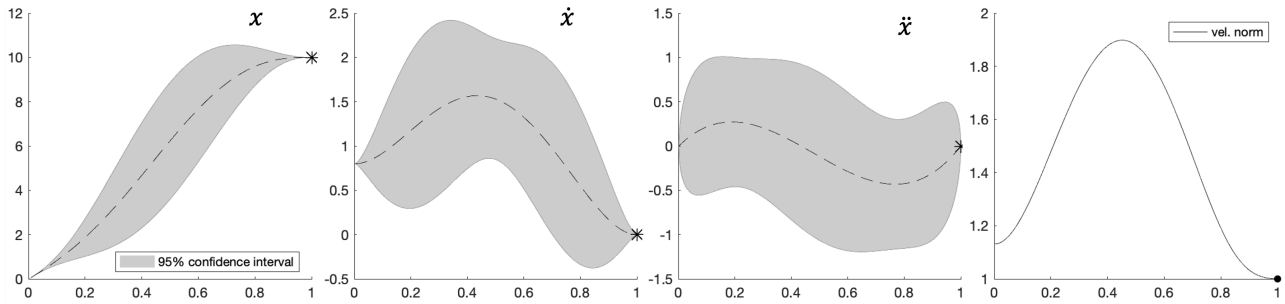


Fig. 2.3 Visual representations of a pseudo-observation-based BD-CA process over time for a single spatial dimension. From left to right: (a) $p(x_n|\mathcal{D}=i,\mathcal{T})$; (b) $p(\dot{x}_n|\mathcal{D}=i,\mathcal{T})$; (c) $p(\ddot{x}_n|\mathcal{D}=i,\mathcal{T})$; (d) velocity norm. All horizontal axes represent time, scaled relative to the terminal time $\mathcal{T}$. Vertical axes denote respectively the value of x, $\dot{x}$, $\ddot{x}_n$, and velocity norm. Dashed lines indicate distribution means. For the generation of this figure, parameters are set as $\tilde{G}=\mathbf{I}$, $\boldsymbol{\Sigma}_i=\mathbf{0}$); in this case the distribution at the endpoint (asterisk) reduces to $p(\mathbf{x}_N|\mathcal{D}=i,\mathcal{T})=\delta_{\mathbf{a}_i}(\mathbf{x}_N)$ with $\delta_{(\cdot)}$ being the Dirac delta function.

where $\mathbf{x}_t=[x_t,\dot{x}_t,\ddot{x}_t,y_t,\dot{y}_t,\ddot{y}_t,z_t,\dot{z}_t,\ddot{z}_t]^\top$ has the same physical meaning as in Section 2.3.1.1 (i.e. position, velocity and acceleration in 3-D Cartesian coordinates),

$$d\mathbf{x}_t=\mathbf{A}\mathbf{x}_t dt+\boldsymbol{\sigma}d\boldsymbol{\beta}_t. \tag{2.16}$$

Here, $\mathbf{A}$ and $\boldsymbol{\sigma}$ are block diagonal matrices with the same number of blocks as in (2.7), but with different block forms. Specifically, $\mathbf{A}_s=[0,1,0;0,0,1;0,0,0]$, and $\boldsymbol{\sigma}_s=[0,0,\sigma]^\top$ $(s=1,2,3)$. The definition of $\boldsymbol{\beta}_t$ is consistent with that in (2.7), meaning it represents a multivariate standard Wiener process. It can be shown that the transition density resulting from this SDE is of the form:

$$p(\mathbf{x}_n|\mathbf{x}_{n-1})=\mathcal{N}(\mathbf{x}_n|\mathbf{F}_h\mathbf{x}_{n-1},\mathbf{Q}_h), \tag{2.17}$$

with $\mathbf{F}_h$ being the state transition matrix, $\mathbf{Q}_h$ the process noise covariance and $h=t_n-t_{n-1}$. In comparison to (2.10), this transition density has no dependency on a destination. When the process noise level is relatively low, (2.17) corresponds to the nearly CA model (also known as the Wiener-process acceleration model). Substituting (2.17) into (2.15) yields

$$\begin{aligned}p(\mathbf{x}_n|\mathbf{x}_{n-1},\mathcal{D}=i,\mathcal{T})&\propto\int\mathcal{N}(\mathbf{a}_i|\tilde{\mathbf{G}}\mathbf{x}_N,\boldsymbol{\Sigma}_i)\mathcal{N}(\mathbf{x}_N|\mathbf{F}_{\mathcal{T}-t_n}\mathbf{x}_n,\mathbf{Q}_{\mathcal{T}-t_n})\mathcal{N}(\mathbf{x}_n|\mathbf{F}_h\mathbf{x}_{n-1},\mathbf{Q}_h)d\mathbf{x}_N\\&\propto\mathcal{N}(\mathbf{a}_i|\tilde{\mathbf{G}}\mathbf{F}_{\mathcal{T}-t_n}\mathbf{x}_n,\tilde{\mathbf{G}}\mathbf{Q}_{\mathcal{T}-t_n}\tilde{\mathbf{G}}^\top+\boldsymbol{\Sigma}_i)\mathcal{N}(\mathbf{x}_n|\mathbf{F}_h\mathbf{x}_{n-1},\mathbf{Q}_h)\\&=\mathcal{N}(\mathbf{x}_n|\mathbf{F}_{i,h}\mathbf{x}_{n-1}+\mathbf{M}_{i,h},\mathbf{Q}_{i,h}),\end{aligned} \tag{2.18}$$

where

$$\begin{aligned}
\mathbf{F}_{i,h} &= \mathbf{Q}_{i,h}\mathbf{Q}_h^{-1}\mathbf{F}_h, \\
\mathbf{M}_{i,h} &= \mathbf{Q}_{i,h}\mathbf{L}\mathbf{a}_i, \\
\mathbf{Q}_{i,h} &= \left(\mathbf{Q}_h^{-1} + \mathbf{L}\tilde{\mathbf{G}}\mathbf{F}_{\mathcal{T}-t_n}\right)^{-1}, \\
\mathbf{L} &= (\tilde{\mathbf{G}}\mathbf{F}_{\mathcal{T}-t_n})^\top(\tilde{\mathbf{G}}\mathbf{Q}_{\mathcal{T}-t_n}\tilde{\mathbf{G}}^\top + \mathbf{\Sigma}_i)^{-1}.
\end{aligned} \tag{2.19}$$

Here (2.18) functions as the transition density for the pseudo-observation process. This process adheres to the law of $p(\mathbf{x}_n|\tilde{\mathbf{y}}_N^i = \mathbf{a}_i, \mathcal{T})$, based on which the state will evolve under the guidance of destination information. Fig. 2.3 gives an example of the marginal distributions obtained according to the above pseudo-observation process where the influence of the endpoint on the state distribution over time is evident. It can also be shown that the limiting distribution, $\lim_{t_N=\mathcal{T}\to\infty} p(\mathbf{x}_N|\mathcal{D}=i,\mathcal{T})$, of a state process having (2.18) as its transition density equates to $\mathcal{N}(\mathbf{a}_i, \mathbf{\Sigma}_i)$ when $\tilde{\mathbf{G}} = \mathbf{I}$. Moreover, setting $\tilde{\mathbf{G}} = \mathbf{I}$ and $\mathbf{\Sigma}_i = \mathbf{0}$ results in a state transition density identical to (2.12), which corresponds a canonical Gaussian bridge [83] terminating at a specific state, given that the endpoint $\mathbf{x}_N$ is certain. It should be stressed that the form of mapping matrix $\tilde{\mathbf{G}}$ depends on what destination-related information is available at hand and thus it is not necessarily equal to an identity matrix; any such matrix can be incorporated into (2.18).

The state transition distributions in equations (2.12) and (2.15) build the destination knowledge into the state dynamics and thus form the basis of BD-based destination-driven (or destination-constrained) motion models. For all nominal endpoints $i = 1, 2, ..., N_D$, a total of $N_\mathcal{D}$ bridges are constructed, with one bridge per endpoint (under the condition $\mathcal{D} = i$). In scenarios where we want the terminal state $\mathbf{x}_N$ at $\mathcal{T}$ as well as $\mathbf{x}_n$ at the current time step t_n to be jointly estimated, the transition model prescribed by (2.12) may be utilised. However, if the main objective is to predict the intended destination as in this chapter with available information on the nominal endpoints (e.g. a certain region/area represented by an ellipsoidal shape), (2.15) can be used to construct a computationally efficient predictor since the hidden state dimension in this case is less than that of the joint estimation scheme (i.e. includes $\mathbf{x}_N$). In Section 2.4.1.2, we present a new intent predictor based on the destination-constrained prior as with (2.15). In comparison to [67], the new predictor requires less computations as it does not estimate the terminal state at t_N. It is constructed using pseudo-observation and therefore the underlying state process is still a Markov process. It also differs from the pseudo-observation-based intent predictors presented in [70] in that it utilises

a destination-constrained state transition density throughout the filtering procedure (although this implies a slightly higher computational burden). Finally, a pseudo-measurement technique for jointly estimating the object state and its destination is presented in [84] based on a linear equality constraint. It dictates that the object follows some straight line to its intended endpoint. Although this simplifies the inference procedure as the condition of arrival time is avoided, it does not capture realistic motion behaviour of several objects of interest (e.g. constraint-free pointing motion in 3-D). On the contrary, the presented stochastic modelling is general and does not impose such restrictive constraints on the target trajectory.

2.3.2 Conditionally linear Gaussian model

The computationally efficient Gaussian model assumes that the change in the object motion (i.e. pointing movements) in any time interval always follows a Gaussian distribution. However, for some irregular movements which cause rapid spatial changes (e.g. jolts in the pointing motion due to perturbations or any external non-intent-driven force; see the example of perturbed pointing trajectories in Fig. 2.4 on page 44), such an assumption is unsuitable and can lead to large inference errors. In order to model such erratic perturbations-induced manoeuvres, we introduce a pure jump process to the original (destination-aware) Gaussian processes. Such formulations are known as jump diffusion models or Markov/Semi-Markov Jump models.

The adopted jump diffusion models retain the Brownian motion as one of the driven noise, and thus they can be considered as a conditionally linear Gaussian system. In particular, when the non-Gaussian pure jump process is given as a condition, the dynamics can be constructed in a standard Gaussian form to ensure the inference tractability.

Such approaches have been extensively adopted in financial modelling to describe the discrete movements [85], and in object tracking field to capture sudden manoeuvres undertaken by the target or induced by external forces [81]. Owing to the clear physical representation and computation tractability, such jump diffusion dynamical models have also been employed in [86] and [69] within a predictive touch system under high levels of perturbations due to road-driving conditions. The approach presented in [86] embedded a self-decay jump process within a Gaussian process to pre-process the highly-perturbed pointing data, with the aim to obtain a smoothed trajectory for the later intent inference task, whereas [69] introduces a jump diffusion model for a unified scheme for destination and state estimation. In this chapter, we mainly discuss the latter recent work given its improved performance.

Since target motion (e.g. pointing gesture movements) impacted by severe external perturbations or fast manoeuvring still exhibits destination-reverting behavior, we formulate the destination-driven model (conditioned on $\mathcal{D} = i$) as a jump diffusion process, adapted from the linear mean-reverting SDE in (2.4):

$$d\mathbf{x}_t = \mathbf{A}(\boldsymbol{\mu}_i - \mathbf{x}_t)dt + \boldsymbol{\sigma} d\boldsymbol{\beta}_t + \mathbf{B}d\mathbf{J}_t, \tag{2.20}$$

where $\mathbf{A}, \boldsymbol{\mu}_i, \boldsymbol{\sigma}, \boldsymbol{\beta}_t$ have the same definitions as in (2.7). If we assume that the jumps only occur at key driving elements of the state (e.g. position for MRD, or acceleration in ERA), the parameter $\mathbf{B} = \text{diag}\{\mathbf{B}_1, \mathbf{B}_2, \mathbf{B}_3\}$ (for 3-D movements) such that $\mathbf{B}_s = [0, 0, 1]^\top$ $(s = 1, 2, 3)$ for ERA and $[0, 1]^\top$ for ERV. The multivariate jump process $\mathbf{J}_t$ here is a compound Poisson process with Gaussian distributed jump size. Specifically, we have $\mathbf{J}_t = \sum_{\tau_k < t} \mathbf{S}_k$, with the jump size $\mathbf{S}_k \in \mathbb{R}^3$ and $\mathbf{S}_k \sim \mathcal{N}(\mathbf{S}_k | \boldsymbol{\mu}_J, \boldsymbol{\Sigma}_J)$. Note that if isotropic distributed jump (i.e. the jump on each direction of the space are identically distributed) is considered, the parameters can be simplified as $\boldsymbol{\mu}_J = \mathbf{0}$ and $\boldsymbol{\Sigma}_J = \sigma_J^2 \mathbf{I}$, where σ_J is defined as the standard deviation of the jump size in any dimension. The jump time τ_k is defined as the arrival times of a homogeneous Poisson process with rate λ_J. It can be showed that the interarrival time satisfies the property $\tau_k - \tau_{k-1} \sim \exp_{\lambda_J}(\cdot)$, where λ_J^{-1} is the mean value of the jump interarrival time. This property defines the transition $p(\tau_k | \tau_{k-1})$ and can be used to sample the jump time from the prior.

Solving SDE (2.20) yields the conditional Markov transition density as follows,

$$p(\mathbf{x}_n | \mathbf{x}_{n-1}, \tau_{n-1:n}) = \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_n^*, \boldsymbol{\Sigma}_n^*), \tag{2.21}$$

with

$$h = t_n - t_{n-1},$$

$$\boldsymbol{\mu}_n^* = \mathbf{F}_{i,h}\mathbf{x}_{n-1} + \mathbf{M}_{i,h} + \sum_{t_{n-1} < \tau_k \leq t_n} \mathbf{F}_{i,t_n - \tau_k} \mathbf{B}\boldsymbol{\mu}_J, \tag{2.22}$$

$$\boldsymbol{\Sigma}_n^* = \mathbf{Q}_{i,h} + \sum_{t_{n-1} < \tau_k \leq t_n} \mathbf{F}_{i,t_n - \tau_k} \mathbf{B}\boldsymbol{\Sigma}_J \mathbf{B}^\top \mathbf{F}_{i,t_n - \tau_k}^\top, \tag{2.23}$$

where $\mathbf{F}_{i,h}$, $\mathbf{M}_{i,h}$ and $\mathbf{Q}_{i,h}$ have been given in (2.11) for the OU-type SDE, and we define the jump time sequence $\tau_{n-1:n}$ as the collection of all jump times that occurred in the interval $(t_{n-1}, t_n]$, i.e. $\tau_{n-1:n} = \bigcup_{t_{n-1} < \tau_k \leq t_n} \{\tau_k\}$. The derivation of (2.22) and (2.23) are straightforward and similar results can be found in [81] and [74]. Subsequently, the sought transition density, when conditioned on the jump time sequence, can be

expressed as

$$p(\mathbf{x}_n|\mathbf{x}_{n-1}, \mathcal{D} = i, \tau_{n-1:n}) = \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_n^*, \boldsymbol{\Sigma}_n^*), \tag{2.24}$$

where $\boldsymbol{\mu}_n^*, \boldsymbol{\Sigma}_n^*$ are given in (2.22) and (2.23).

2.4 Destination prediction

The overall system is described by the destination-driven motion models in Section 2.3 and the linear Gaussian observation model in (2.2). Next, we introduce various destination inference algorithms to calculate the sought probabilities $p(\mathcal{D} = i|\mathbf{y}_{1:n})$, $i = 1, 2, ..., N_D$, in (2.1). As shown below, the intent inference routine complexity is dependent on the employed motion model. For instance, a linear Gaussian set-up leads to a simple and computationally efficient Kalman filer-based predictor for the destination inference task.

Recall from (2.1) that the key to sequentially infer the probability of a nominal endpoint i being the intended destination is to estimate the likelihood $p(\mathbf{y}_{1:n}|\mathcal{D} = i)$. Furthermore, this likelihood can be recursively expanded according to *prediction error decomposition* (PED) [87] given by

$$p(\mathbf{y}_{1:n}|\mathcal{D} = i) = p(\mathbf{y}_n|\mathbf{y}_{1:n-1}, \mathcal{D} = i)p(\mathbf{y}_{1:n-1}|\mathcal{D} = i). \tag{2.25}$$

This sequential likelihood estimation serves as the basis of online Bayesian intent predictor as it only requires the evaluation of predictive likelihood $p(\mathbf{y}_n|\mathbf{y}_{1:n-1}, \mathcal{D} = i)$ at each time instant. In this section, we discuss the strategy to compute this predictive likelihood for the various models introduced in Section 2.3.

2.4.1 Linear Gaussian systems

Although the destination-driven models in Section 2.3.1 are designed to have linear Gaussian transition densities, it is worth noting that only the OU-based models possess a purely linear Gaussian transition $p(\mathbf{x}_n|\mathbf{x}_{n-1}, \mathcal{D} = i)$. In contrast, the BD-based model's transition $p(\mathbf{x}_n|\mathbf{x}_{n-1}, \mathcal{D} = i, \mathcal{T})$ is linear Gaussian only when conditioned on a specific static parameter — the arrival time $\mathcal{T}$. Assuming a linear Gaussian observation model (e.g. for an off-the-shelf gesture tracker) in (2.2), the standard Kalman filter is sufficient for carrying out recursive state estimation (in both the Bayesian optimal and minimum mean square error sense [62]) and intent inference that computes the

exact $p(\mathcal{D}=i|\mathbf{y}_{1:n})$ when tackling the OU-based models or the arrival time-conditioned BD-based models.

2.4.1.1 OU-based intent predictors

Given that the transition density for the OU-based model in (2.10) and the observation function in (2.2) are both linear Gaussian, the posterior and predictive distribution of the target state can be explicitly represented as normal distributions. Specifically,

$$p(\mathbf{x}_n|\mathbf{y}_{1:n},\mathcal{D}=i)=\mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}^i_{n|n},\mathbf{C}^i_{n|n}), \tag{2.26}$$
$$p(\mathbf{x}_n|\mathbf{y}_{1:n-1},\mathcal{D}=i)=\mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}^i_{n|n-1},\mathbf{C}^i_{n|n-1}). \tag{2.27}$$

The predictive likelihood can be computed as follows,

$$p(\mathbf{y}_n|\mathbf{y}_{1:n-1},\mathcal{D}=i)=\int p(\mathbf{y}_n|\mathbf{x}_n)p(\mathbf{x}_n|\mathbf{y}_{1:n-1},\mathcal{D}=i)d\mathbf{x}_n. \tag{2.28}$$

With the observation model in (2.2), this results in a Gaussian likelihood:

$$\begin{aligned} p(\mathbf{y}_n|\mathbf{y}_{1:n-1},\mathcal{D}=i)&=\mathcal{N}(\mathbf{y}_n|\boldsymbol{\mu}^i_{y_n},\mathbf{C}^i_{y_n})\\ \boldsymbol{\mu}^i_{y_n}&=\mathbf{H}\boldsymbol{\mu}^i_{n|n-1},\\ \mathbf{C}^i_{y_n}&=\mathbf{H}\mathbf{C}^i_{n|n-1}\mathbf{H}^\top+\mathbf{V}_n. \end{aligned} \tag{2.29}$$

To compute $\boldsymbol{\mu}^i_{n|n-1}$ and $\mathbf{C}^i_{n|n-1}$ at each time step, the standard Kalman filter is required to estimate the state recursively, summarised as follows,

$$p(\mathbf{x}_n|\mathbf{y}_{1:n-1},\mathcal{D}=i)=\int p(\mathbf{x}_{n-1}|\mathbf{y}_{1:n-1},\mathcal{D}=i)p(\mathbf{x}_n|\mathbf{x}_{n-1},\mathcal{D}=i)d\mathbf{x}_{n-1}, \tag{2.30}$$
$$p(\mathbf{x}_n|\mathbf{y}_{1:n},\mathcal{D}=i)\propto p(\mathbf{y}_n|\mathbf{x}_n)p(\mathbf{x}_n|\mathbf{y}_{1:n-1},\mathcal{D}=i). \tag{2.31}$$

The corresponding matrix description is

$$\begin{aligned} \boldsymbol{\mu}^i_{n|n-1}&=\mathbf{F}_{i,h}\boldsymbol{\mu}^i_{n-1|n-1}+\mathbf{M}_{i,h},\\ \mathbf{C}^i_{n|n-1}&=\mathbf{F}_{i,h}\mathbf{C}^i_{n-1|n-1}\mathbf{F}^\top_{i,h}+\mathbf{Q}_{i,h},\\ \mathbf{K}&=\mathbf{C}^i_{n|n-1}\mathbf{H}^\top{\mathbf{C}^i_{\mathbf{y}_n}}^{-1},\\ \boldsymbol{\mu}^i_{n|n}&=\boldsymbol{\mu}^i_{n|n-1}+\mathbf{K}(\mathbf{y}_n-\boldsymbol{\mu}^i_{y_n}),\\ \mathbf{C}^i_{n|n}&=(\mathbf{I}-\mathbf{K}\mathbf{H})\mathbf{C}^i_{n|n-1}, \end{aligned} \tag{2.32}$$

where $\mathbf{F}_{i,h}, \mathbf{M}_{i,h}, \mathbf{Q}_{i,h}$ are given in (2.11) for OU-based model. The above equations outline the the computation of predictive likelihood for a single time step. The likelihood for each destination being the intended one can then be sequentially evaluated with (2.1), (2.25) and (2.29).

2.4.1.2 BD-based intent predictor using pseudo-observation formulation

In principle, BD-based intent predictors, including those in [68, 70] and the new approach introduced here, all utilise (2.1) and (2.25) for inferring the target destination from the available noisy sensory observations. However, for the BD approach proposed here, the closed-form likelihood $p(\mathbf{y}_{1:n}|\mathcal{D} = i, \mathcal{T})$ is conditioned not only on the destination-specific parameters (in this case, $\mathbf{a}_i$ and $\mathbf{\Sigma}_i$), but also on $\mathcal{T}$, which represents the arrival time at the destination. Given the additional conditioning on an unknown arrival time $\mathcal{T}$, (2.25) needs to be revised as follows:

$$p(\mathbf{y}_{1:n}|\mathcal{D} = i, \mathcal{T}) = p(\mathbf{y}_n|\mathbf{y}_{1:n-1}, \mathcal{D} = i, \mathcal{T})p(\mathbf{y}_{1:n-1}|\mathcal{D} = i, \mathcal{T}), \tag{2.33}$$

based on which $p(\mathbf{y}_{1:n}|\mathcal{D} = i)$ can be obtained via

$$p(\mathbf{y}_{1:n}|\mathcal{D} = i) = \int p(\mathbf{y}_{1:n}|\mathcal{D} = i, \mathcal{T})p(\mathcal{T}|\mathcal{D} = i)d\mathcal{T}, \tag{2.34}$$

where $p(\mathcal{T}|\mathcal{D} = i)$ is the prior distribution on the unknown arrival time. In general, the above integration is not analytically tractable and numerical approximation can be implemented. This is especially viable since the arrival time is a one-dimensional quantity (and thereby the integral). In this chapter, we will adopt the same quadrature approximation scheme as in [68] for obtaining (2.34).

Henceforth, the aim is to compute the arrival time-conditioned likelihood (i.e. the *unknown* arrival time is treated as if it is available). We illustrate how to develop an intent predictor based on the destination-constrained process defined in Section 2.3.1.2. Given observations up to t_n, the $\mathcal{T}$-conditioned likelihood term of interest can be expressed by

$$\begin{aligned} &p(\mathbf{y}_n|\mathbf{y}_{1:n-1}, \mathcal{D} = i, \mathcal{T}) \\ &= \int p(\mathbf{y}_n|\mathbf{x}_n)p(\mathbf{x}_n|\mathbf{x}_{n-1}, \mathcal{D} = i, \mathcal{T})p(\mathbf{x}_{n-1}|\mathbf{y}_{1:n-1}, \mathcal{D} = i, \mathcal{T})d\mathbf{x}_{n-1}d\mathbf{x}_n, \end{aligned} \tag{2.35}$$

where the first component in the integral is the observation density, the second component is the destination-constrained state transition density as defined in (2.15) and the last component is a filtering distribution obtained at time t_{n-1}. Next, we outline

how to calculate $p(\mathbf{y}_n|\mathbf{y}_{1:n-1}, \mathcal{D}=i, \mathcal{T})$ at each time step for a linear Gaussian dynamic system. For simplicity and without loss of generality, we use the same underlying state model as with (2.17) with destination information incorporated via (2.13). This implies the availability of the destination-conditioned transition density in equation (2.18). With a linear Gaussian observation model in (2.2), we have

$$p(\mathbf{y}_n|\mathbf{x}_n) = \mathcal{N}(\mathbf{y}_n|\mathbf{H}\mathbf{x}_n, \mathbf{V}_n), \tag{2.36}$$

where $\mathbf{H}$ is the observation matrix and $\mathbf{V}_n$ is the measurement noise covariance matrix. As a result, the filtering distribution $p(\mathbf{x}_{n-1}|\mathbf{y}_{1:n-1}, \mathcal{D}=i, \mathcal{T})$ at the previous time step t_{n-1} can be obtained using a standard Kalman filter in which (2.18) is used as the state transition density. Assuming at t_n, we have obtained the filtering distribution provided by the Kalman filter associated with the i-th nominal endpoint from the last time step t_{n-1} as

$$p(\mathbf{x}_{n-1}|\mathbf{y}_{1:n-1}, \mathcal{D}=i, \mathcal{T}) = \mathcal{N}(\mathbf{x}_{n-1}|\boldsymbol{\mu}^i_{n-1|n-1}, \mathbf{C}^i_{n-1|n-1}), \tag{2.37}$$

with $\boldsymbol{\mu}^i_{n-1|n-1}$ and $\mathbf{C}^i_{n-1|n-1}$ being the mean and covariance respectively. By substituting (2.36), (2.18) and (2.37) into (2.35), the sought likelihood can be shown to be

$$\begin{aligned}
&p(\mathbf{y}_n|\mathbf{y}_{1:n-1}, \mathcal{D}=i, \mathcal{T}) \\
=&\int \mathcal{N}(\mathbf{y}_n|\mathbf{H}\mathbf{x}_n, \mathbf{V}_n)\mathcal{N}(\mathbf{x}_n|\mathbf{F}_{i,h}\mathbf{x}_{n-1} + \mathbf{M}_{i,h}, \mathbf{Q}_{i,h})\mathcal{N}(\mathbf{x}_{n-1}|\boldsymbol{\mu}^i_{n-1|n-1}, \mathbf{C}^i_{n-1|n-1})d\mathbf{x}_{n-1}d\mathbf{x}_n \\
=&\mathcal{N}\Big(\mathbf{y}_n \mid \mathbf{H}(\mathbf{F}_{i,h}\boldsymbol{\mu}^i_{n-1|n-1} + \mathbf{M}_{i,h}),\ \mathbf{H}(\mathbf{F}_{i,h}\mathbf{C}^i_{n-1|n-1}\mathbf{F}^\top_{i,h} + \mathbf{Q}_{i,h})\mathbf{H}^\top + \mathbf{V}_n\Big),
\end{aligned} \tag{2.38}$$

where $\mathbf{F}_{i,h}, \mathbf{M}_{i,h}, \mathbf{Q}_{i,h}$ is given in (2.19) for the BD model. The above calculation can be further simplified by noticing that

$$\begin{aligned}
\boldsymbol{\mu}^i_{n|n-1} &= \mathbf{F}_{i,h}\boldsymbol{\mu}^i_{n-1|n-1} + \mathbf{M}_{i,h} \\
\mathbf{C}^i_{n|n-1} &= \mathbf{F}_{i,h}\mathbf{C}^i_{n-1|n-1}\mathbf{F}^\top_{i,h} + \mathbf{Q}_{i,h}
\end{aligned} \tag{2.39}$$

are actually the mean and covariance of the intermediate distribution $p(\mathbf{x}_n|\mathbf{y}_{1:n-1}, \mathcal{D}=i, \mathcal{T}) = \mathcal{N}(\mathbf{x}_{n-1}|\boldsymbol{\mu}^i_{n|n-1}, \mathbf{C}^i_{n|n-1})$ obtained at the Kalman prediction step. As a result, there is no need to re-calculate these two quantities twice.

Combining (2.33), (2.38) and (2.34), $p(\mathbf{y}_{1:n}|\mathcal{D}=i)$ can be evaluated sequentially when new measurements become available. To complete the intent prediction algorithm, the above calculation needs to be performed for each nominal endpoint $i = 1, 2, ..., N_D$. Furthermore, when a quadrature approximation scheme is used, (2.38) needs to be

Algorithm 1: BD-based intent predictor

Input: Observations: $\{\mathbf{y}_{1:N}\}$, Pseudo-observations: $\{\mathbf{a}_i, \mathbf{\Sigma}_i\}_{1\leq i\leq N_{\mathcal{D}}}$, $\mathbb{T} = \{\mathcal{T}_q\}_{1\leq q\leq N_{\mathcal{T}}}$.
Initialisation: $N_{\mathcal{D}} \times N_{\mathcal{T}}$ Kalman filters, each initialised with mean $\boldsymbol{\mu}_{0|0}^{i,q}$ and covariance $\mathbf{C}_{0|0}^{i,q}$.
for $n = 1 : N$ **do**
  for $i = 1 : N_D$ **do**
    for $\mathcal{T}_q \in \mathbb{T}$ **do**
      Construct the intent-driven transition density $p(\mathbf{x}_n|\mathbf{x}_{n-1}, \mathcal{D} = i, \mathcal{T}_q)$ via (2.18).
      Standard Kalman prediction to obtain $\boldsymbol{\mu}_{n|n-1}^{i,q}$ and $\mathbf{C}_{n|n-1}^{i,q}$ via (2.39).
      Standard Kalman update to obtain $\boldsymbol{\mu}_{n|n}^{i,q}$ and $\mathbf{C}_{n|n}^{i,q}$.
      Compute: $l_n^{i,q} = p(\mathbf{y}_n|\mathbf{y}_{1:n-1}, \mathcal{D} = i, \mathcal{T}_q)$ via (2.38).
      Update $\mathcal{T}_q$-conditioned likelihood via (2.33): $p(\mathbf{y}_{1:n}|\mathcal{D} = i, \mathcal{T}_q) = L_n^{i,q} = L_{n-1}^{i,q} \times l_n^{i,q}$
    end
    Approximate $p(\mathbf{y}_{1:n}|\mathcal{D} = i)$ numerically using $\{L_n^{i,q}, q = 1, 2, \ldots, N_{\mathcal{T}}\}$.
  end
  Obtain destination posterior at t_n: $p(\mathcal{D} = i|\mathbf{y}_{1:n}) \approx \frac{p(\mathbf{y}_{1:n}|\mathcal{D}=i)\times p(\mathcal{D}=i)}{\sum_{j=1}^{N_D} p(\mathbf{y}_{1:n}|\mathcal{D}=j)\times p(\mathcal{D}=j)}$.
end

evaluated at each quadrature point of $\mathbb{T} = \{\mathcal{T}_q, q = 1, 2, \ldots, N_{\mathcal{T}}\}$. A detailed implementation note is summarised in Algorithm 1. It is noted that a guidance on the choice number of quadrature points for BD methods can be found in [68].

At each time step n, the intent inference procedure presented in Algorithm 1 requires $N_D N_{\mathcal{T}}$ Kalman filter runs. This is the same as the intent inference procedure of the original BD methods, as described in Algorithm 2 of [68]. However, each Kalman filter run in the latter has a higher computational demand due to the estimation of the augmented joint states of $\mathbf{x}_n$ and $\mathbf{x}_N$, making our Algorithm 1 more computationally efficient overall. Nonetheless, it's worth noting that the intent inference approach in Algorithm 1 of [70] remains superior in computational efficiency compared to our Algorithm 1. This advantage stems from its exploitation of specific mathematical properties to bypass the Kalman filter predictions (2.39) for each BD model driven by various destinations. For further details, refer to [70].

2.4.2 Intent predictors for jump diffusion models

The jump diffusion model introduced in Section 2.3.2 is constructed in a conditionally Gaussian form as shown in (2.21). Specifically, the transition density from time t to $t+h$ is a Gaussian density if the nonlinear component jump time sequence $\tau_{t:t+h}$ is given as a condition. Thus an efficient strategy would be estimating $\tau_{t:t+h}$ using a Monte Carlo approach, then for each sample of $\tau_{t:t+h}$, $p(\mathbf{x}_{t+h,i}|\mathbf{x}_{t,i}, \tau_{t:t+h})$ is retained as Gaussian form so that the standard Kalman filter can be employed to carry out the estimation. This strategy, known as Rao-Blackwellisation [27, 45], aims to enhance estimation accuracy (by reducing estimator variance) by incorporating analytical computations into the Monte Carlo methods as much as possible.

Additionally, the particle filter used here is known as the variable rate particle filter (VRPF) [81, 74, 88]. The key difference between the VRPF and the standard particle filter lies in the nature of the time series variables of interest, which are set to be inferred through sampling. In the VRPF, these time series variables have different time index from when the observations are made, resulting in sampling that occurs at variable rates in relation to when observations are taken. In contrast, the standard particle filter samples the state precisely at the observation times. The use of the VRPF facilitates the detection of changepoints in the dynamics between consecutive observations. This approach enables more complicated dynamic models where characteristics, such as the transition density's parametric form, can vary at a sequence of changepoint times (i.e. the jump times), and both the variation and change times are random and can be estimated from unsynchronised observations.

When $N_{\mathcal{D}}$ possible destinations are considered, the same number of particle filters are required, each with $N_{\mathcal{P}}$ particles for a particular nominal endpoint. Here, we allow the $N_{\mathcal{D}}$ different particle filters to share the same sample set of jump times. This not only reduces the inference computational complexity, but can also circumvent spurious large differences between the likelihoods of the various endpoints, induced by individual sample outlier(s). Nonetheless, this particular consideration is not expected to noticeably impact the intent prediction performance since the aim in this chapter is *not* to accurately estimate the object state or the individual destination likelihood $p(\mathbf{y}_{1:n}|\mathcal{D} = i)$. Instead, the focus is on comparing the likelihoods for all nominal destinations, calculated under the same conditions, in order to determine the intended endpoint from the observed motion. At time t_n, each variable rate particle filter stores the samples $\tau_{0:n}^{(p)}$ $(p = 1, 2, ..., N_{\mathcal{P}})$, the *normalised* weight $\omega_n^{(p,i)}$, the mean $\boldsymbol{\mu}_{n|n}^{(p,i)}$ and covariance $\mathbf{C}_{n|n}^{(p,i)}$ for Gaussian density $p(\mathbf{x}_n|\mathbf{y}_{1:n}, \tau_{0:n}^{(p)}, \mathcal{D} = i)$. Subsequently, the

empirical estimations for jump time and state can be described as follows,

$$p(\tau_{0:n}|\mathbf{y}_{1:n}, \mathcal{D}=i) \approx \sum_{p=1}^{N_{\mathcal{P}}} \omega_n^{(p,i)} \delta_{\tau_{0:n}^{(p)}}(\tau_{0:n}), \tag{2.40}$$

$$p(\mathbf{x}_n|\mathbf{y}_{1:n}, \mathcal{D}=i) \approx \sum_{p=1}^{N_{\mathcal{P}}} \omega_n^{(p,i)} \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_{n|n}^{(p,i)}, \mathbf{C}_{n|n}^{(p,i)}). \tag{2.41}$$

Accordingly, the predictive likelihood can be approximated as (see e.g. [43, 46])

$$\begin{aligned} p(\mathbf{y}_{n+1}|\mathbf{y}_{1:n}, \mathcal{D}=i) &= \int p(\mathbf{y}_{n+1}|\mathbf{y}_{1:n}, \tau_{0:n+1}, \mathcal{D}=i) p(\tau_{n:n+1}|\tau_{0:n}) p(\tau_{0:n}|\mathbf{y}_{1:n}, \mathcal{D}=i) d\tau_{0:n+1} \\ &\approx \sum_{p=1}^{N_{\mathcal{P}}} \tilde{\omega}_{n+1}^{(p,i)}, \end{aligned} \tag{2.42}$$

where the *updated* weight $\tilde{\omega}_{n+1}^{(p,i)}$, in the bootstrap particle filter setting, is defined as

$$\tilde{\omega}_{n+1}^{(p,i)} = \omega_n^{(p,i)} p(\mathbf{y}_{n+1}|\mathbf{y}_{1:n}, \tau_{0:n+1}^{(p)}, \mathcal{D}=i). \tag{2.43}$$

The detailed derivation of (2.43) can be found in [69]. The new jump time samples $\tau_{n:n+1}^{(p)}$, in the corresponding (bootstrap) setup, are propagated according to the homogeneous Poisson process with rate λ_J, whose transition is described in Section 2.3.2,

$$\tau_{n:n+1}^{(p)} \sim p(\tau_{n:n+1}|\tau_{0:n}^{(p)}). \tag{2.44}$$

More details on the sampling procedure of $\tau_{n:n+1}$ according to (2.44) can be found in [69]. Subsequently, the *normalised* weight can be computed with $\tilde{\omega}_{n+1}^{(p,i)}$ in (2.43) as

$$\omega_{n+1}^{(p,i)} = \frac{\tilde{\omega}_{n+1}^{(p,i)}}{\sum_{p=1}^{N_{\mathcal{P}}} \tilde{\omega}_{n+1}^{(p,i)}}. \tag{2.45}$$

Similar to (2.28), the $p(\mathbf{y}_{n+1}|\mathbf{y}_{1:n}, \tau_{0:n+1}^{(p)}, \mathcal{D}=i)$ in (2.43) can be computed in a closed form with the stored mean $\boldsymbol{\mu}_{n|n}^{(p,i)}$ and covariance $\mathbf{C}_{n|n}^{(p,i)}$, i.e.

$$p(\mathbf{y}_{n+1}|\mathbf{y}_{1:n}, \tau_{0:n+1}^{(p)}, \mathcal{D}=i) = \mathcal{N}(\mathbf{y}_{n+1}|\mathbf{H}\boldsymbol{\mu}_{n+1|n}^{(p,i)}, \mathbf{H}\mathbf{C}_{n+1|n}^{(p,i)}\mathbf{H}^\top + \mathbf{V}_{n+1}), \tag{2.46}$$

where

$$\begin{aligned}\boldsymbol{\mu}_{n+1|n}^{(p,i)} =& \mathbf{F}_{i,h}\boldsymbol{\mu}_{n|n}^{(p,i)} + \mathbf{M}_{i,h} + \sum_{t_n<\tau_k^{(p)}\leq t_{n+1}} \mathbf{F}_{i,t_{n+1}-\tau_k^{(p)}}\mathbf{B}\boldsymbol{\mu}_J, \\ \mathbf{C}_{n+1|n}^{(p,i)} =& \mathbf{F}_{i,h}\mathbf{C}_{n|n}^{(p,i)}\mathbf{F}_{i,h}^{\top} \\ &+ \sum_{t_n<\tau_k^{(p)}\leq t_{n+1}} \mathbf{F}_{i,t_{n+1}-\tau_k^{(p)}}\mathbf{B}\boldsymbol{\Sigma}_J\mathbf{B}^{\top}\mathbf{F}_{i,t_{n+1}-\tau_k^{(p)}}^{\top} + \mathbf{Q}_{i,h}.\end{aligned} \tag{2.47}$$

In order to update the stored density mean $\boldsymbol{\mu}_{n+1|n+1}^{(p,i)}$ and covariance $\mathbf{C}_{n+1|n+1}^{(p,i)}$, the following standard Kalman filter updated steps are required:

$$\begin{aligned}\boldsymbol{\mu}_{n+1|n+1}^{(p,i)} &= \boldsymbol{\mu}_{n+1|n}^{(p,i)} + \mathbf{K}(\mathbf{y}_n - \mathbf{H}\boldsymbol{\mu}_{n+1|n}^{(p,i)}), \\ \mathbf{C}_{n+1|n+1}^{(p,i)} &= (\mathbf{I} - \mathbf{K}\mathbf{H})\mathbf{C}_{n+1|n}^{(p,i)}, \\ \mathbf{K} &= \mathbf{C}_{n+1|n}^{(p,i)}\mathbf{H}^{\top}(\mathbf{H}\mathbf{C}_{n+1|n}^{(p,i)}\mathbf{H}^{\top} + \mathbf{V}_{n+1})^{-1}.\end{aligned} \tag{2.48}$$

The procedure described above constitutes a complete variable rate particle filtering procedure for a single time step. The overall intent prediction algorithm is summarised as Algorithm 2.

2.5 Results

The performance of the discussed destination prediction methods is assessed below using free hand pointing gestures recorded in vehicles instrumented with a predictive touch system as in Fig. 2.1. This system used the off-the-shelf sensor, Leap Motion [64], which tracks hand and finger positions in 3-D during the pointing-selection tasks, at a rate exceeding 30 Hz. The introduction of this pointing gesture tracker is provided in Section 2.1.1. The utilised dataset contains 95 trajectories pertaining to four participants whilst undertaking pointing-selection tasks under various road and driving conditions. Here, we divide this data into two sets:

i). Dataset A with all 95 pointing tracks; this allows us to perform a comprehensive comparison between different algorithms for various levels of present perturbations (e.g. static, motorway driving and off-road driving).

ii). Dataset B with 10 trajectories when the user input was subjected to severe level of noise due to driving on a badly maintained road or off-road driving. This dataset is a subsect of Dataset A and was collected in a Land Rover. It is particularly

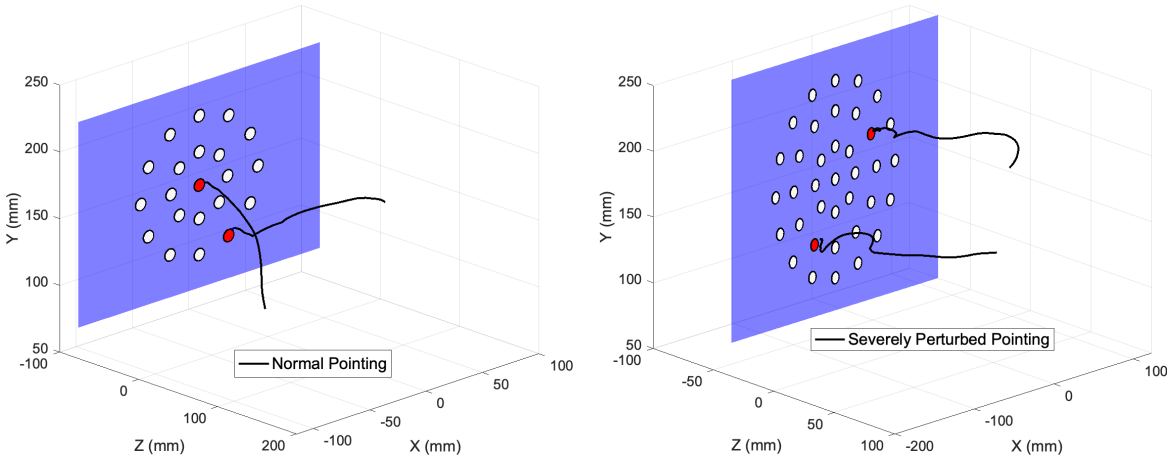
Algorithm 2: Intent inference with the jump model

Initialisation: Create $N_{\mathcal{D}}$ variable rate particle filters, each with $N_{\mathcal{P}}$ particles. For each particle p and each destination label i, initialise the mean as $\boldsymbol{\mu}_{0|0}^{(p,i)}$, covariance as $\mathbf{C}_{0|0}^{(p,i)}$, and weight as $\omega_0^{(p,i)}$.
for $n = 1 : N$ **do**
  for *particles* $p = 1 : N_{\mathcal{P}}$ **do**
    Sample the jump time sequence from prior $\tau_{n-1:n}^{(p)}$ from (2.44).
  end
  for *endpoints* $i = 1 : N_{\mathcal{D}}$ **do**
    if *Resample* **then**
      Resample particles and set weights $\omega_{n-1}^{(p,i)} = 1/N_{\mathcal{P}}$.
    end
    for *particles* $p = 1 : N_{\mathcal{P}}$ **do**
      Evaluate the predictive mean $\boldsymbol{\mu}_{n|n-1}^{(p,i)}$ and covariance $\mathbf{C}_{n|n-1}^{(p,i)}$ via (2.47).
      Calculate the *updated* weight $\tilde{\omega}_n^{(p,i)}$ according to (2.43)(2.46).
      Update the mean $\boldsymbol{\mu}_{n|n}^{(p,i)}$ and covariance $\mathbf{C}_{n|n}^{(p,i)}$ via (2.48).
    end
    Compute the predictive likelihood $p(\mathbf{y}_n|\mathbf{y}_{1:n-1}, \mathcal{D} = i)$ from (2.42).
    Calculate the *normalised* weight $\omega_n^{(p,i)}$ according to (2.45).
    Calculate likelihood $p(\mathbf{y}_{1:n}|\mathcal{D} = i)$ in (2.25).
  end
  Determine endpoint probability: $p(\mathcal{D} = i|\mathbf{y}_{1:n})$ in (2.1).
end

relevant to examine the outcome of the algorithms that incorporate a jump process, i.e. employ jump diffusion models.

During the above interaction tasks, an experimental user interface with multiple selectable circular icons was displayed on a touchscreen mounted to the car dashboard. The number of selectable icons is $N_D = 21$ for Dataset A, and $N_D = 37$ for Dataset B. Two typical pointing trajectories of each dataset are presented in Fig. 2.4. Similar to the common ISO 9241 pointing task[2], often referred to as Fitt's law task, one randomly

[2]The ISO 9241-411 standard [89] specifically outlines two types of point-select tasks for assessing the performance of non-keyboard input devices. These are: 1) one-directional point-select tests, which focus on movements along a single axis; and 2) multi-directional point-select tests, which involve more intricate movements in various directions. The main metric for evaluation in both tests is 'throughput', measured in bits per second, capturing both the speed and accuracy of the input device. An example of a multi-directional tapping task will be discussed in Section 3.6.1 in a subsequent chapter. See the original standard [89] for further details.



(a) Dataset A (21 icons) (b) Dataset B (37 icons)

Fig. 2.4 Example trajectories of collected real pointing trajectories.

chosen GUI item is highlighted at a time and the user is expected to select it. Identical uniform prior is placed on all of the interface items, i.e. $p(\mathcal{D} = i) = 1/N_{\mathcal{D}}$ for all nominal endpoints $i = 1, 2, ..., N_D$ in order for the results to be comparable to those in previous work.

2.5.0.1 Performance metrics and discussions

In this chapter, we use two metrics to evaluate the predictor's performance: aggregate inference success and timely successful prediction over the pointing duration. Both metrics employ a Maximum A Posteriori (MAP) criterion, which selects the most probable icon as follows:

$$\hat{I}_n = \underset{i=1,2,...,N_D}{\arg\max} \; p(\mathcal{D} = i \mid \mathbf{y}_{1:n}),$$

Here, $\hat{I}_n$ is the index of the most probable destination estimated at the n-th time step. More specifically, the aggregate inference success is defined as the proportion of the total pointing gesture duration (in time), from its start at t_0 until touching the display surface at time t_N, during which the predictor correctly inferred the true index of the endpoint, $I_{\text{True}} \in \{1, 2, ..., N_D\}$. The timely successful prediction captures the percentage of the correct prediction over all tested dataset as a function of the percentage of pointing task duration, thus indicating how early the predictor assigns the highest probability for the correct destination.

These two metrics have been deemed valuable in several prior studies, e.g. [90, 67, 7, 68]. However, an important observation is that both metrics, computed using the strict MAP estimate $\hat{I}_n$ in (2.49), don't consider the confidence level of the MAP estimate. This means that a destination predictor that is significantly off-mark (e.g. where $p(\mathcal{D} = I_{\text{True}} \mid \mathbf{y}_{1:n}) = 0$) and one that's just narrowly incorrect (e.g. where $p(\mathcal{D} = I_{\text{True}} \mid \mathbf{y}_{1:n})$ is just 1% lower than the $p(\mathcal{D} = \hat{I}_n \mid \mathbf{y}_{1:n})$) are equivalently labeled as erroneous by these metrics. While this may seem like a limitation, in practical scenarios, intent predictor often render a single definitive outcome without conveying the underlying uncertainty. For instance, the user of a predictive touch only need an accurate prediction of the intended icon, rather than understanding the confidence or uncertainty of the prediction. In this context, a predictor that's either slightly or significantly wrong is equally unhelpful.

The question then arises: is there a metric that can capture both prediction uncertainty and the real-world applicability of an intent predictor? The answer is affirmative. In fact, one could incorporate prediction uncertainty to enhance the reliability of an intent predictor's output. Ideally, a predictive touch system should quickly and confidently select the right icon, avoiding mistakes. An intent predictor, thus, should make a selection only when there's high certainty. Therefore, a possible metric, which accounts for both prediction uncertainty and the definitive decision making in the real-world intent predictor, could evaluate the accuracy of these 'confident' predictions (this can be defined as the proportion of dataset where the 'confident' predictions is correct) and the time taken to reach such a decision. However, this presents a new challenge: defining 'confidence' in a prediction. This definition can vary (for example, where $p(\mathcal{D} = \hat{I}_n \mid \mathbf{y}_{1:n})$ first hits a threshold like 0.75), and the performance under the metric would be influenced by this choice. Therefore, this section adheres to the precedent set in [90, 67, 7, 68], using aggregate inference success and timely successful prediction metrics. These metrics are free from such arbitrary determinations and provide a balanced assessment of different algorithms' intent inference capabilities.

2.5.0.2 Parameter selections

All models examined in this section have their parameters approximately tuned by hand to give a satisfactory level of performance in a range of data examples. This hand-tuning process comprises two stages: coarse tuning and fine tuning. Both are akin to grid searching but vary in their discretisation levels.

- Coarse Tuning: This preliminary phase involves examining a set of parameter values that span a broad spectrum. The primary objective here is to identify a general parameter space that yields reasonable performance.
- Fine Tuning: Post the coarse tuning, this phase narrows its focus, probing deeper into specific regions near the promising parameter values identified earlier. The aim here is to pinpoint those 'local optima' where the parameter values produce superior performance.

During the coarse tuning phase, the inherent physical interpretations of our model, an advantage of our stochastic process with physical meaning, may guide the exclusion of certain parameter ranges. For instance, considering the linear Gaussian mean-reverting model discussed in Section 2.3.1.1 on page 26: an increased reversion parameter, η, results in a more forceful tendency towards the endpoint. This necessitates an increased damping factor ρ (and possibly γ) to maintain reasonable finger speed at the touchpoint/destination. As such, emphasis is placed on configurations where both the reversion parameter and the damping factor rise in tandem. Furthermore, the noise parameter σ can influence the model's adaptability to varying datasets, such as static pointing versus perturbed pointing. A large σ leads to high motion uncertainty, making it more likely to capture the perturbed pointing behaviour. Consequently, given the mix of perturbed and static data in the comprehensive test dataset, multiple 'local optima' for σ might emerge that cater to an overall good performance. This calls for a meticulous exploration of various σ values to avoid overlooking an optimal parameter setting.

However, it's imperative to highlight certain limitations associated with tuning parameters based purely on their physical interpretation:

1. The utility of physical meaning in parameter selection is largely confined to specific parameter combinations. As an illustrative example, there isn't an evident correlation between the noise parameter σ and damping factor ρ that could be harnessed for tuning purposes.

2. While physical interpretations can be insightful, the overall complexity of the inference algorithm often surpasses what can be inferred from mere intuition. While these insights may inform parameter choices for better motion representation or improved tracking, they don't always translate to optimal destination predictions, or the other way around. For instance, a model with low noise, indicative of a straight-line motion, might not accurately represent the actual trajectory of an

Table 2.1 Linear Gaussian models parameters and overall prediction performance for 95 tracks.

Models	Parameter Values	Success Rates
ERV ([67])	$\eta = 55$, $\rho = 15$, $\sigma = 3000$	62.9%
ERA ([69])	$\eta = 1150$, $\rho = 320$, $\gamma = 29$, $\sigma = 1.7 \times 10^4$	63.4%
BD-CV ([68])	$\sigma_{\text{CV}} = 650$, $\sigma^{\mathcal{D}}_{\text{pos}} = 1.5$, $\sigma^{\mathcal{D}}_{\text{vel}} = 100$	64.4%
BD-CV ([70])	$\sigma_{\text{CV}} = 650$, $\sigma^{\mathcal{D}}_{\text{pos}} = 1.5$, $\sigma^{\mathcal{D}}_{\text{vel}} = 100$	65.4%
BD-CA ([70])	$\sigma_{\text{CA}} = 9400$, $\sigma^{\mathcal{D}}_{\text{pos}} = 1.5$, $\sigma^{\mathcal{D}}_{\text{vel}} = 50$, $\sigma^{\mathcal{D}}_{\text{acc}} = 1500$	68.3%
BD-CV (this chapter)	$\sigma_{\text{CV}} = 650$, $\sigma^{\mathcal{D}}_{\text{pos}} = 1.5$, $\sigma^{\mathcal{D}}_{\text{vel}} = 100$	65.2%
BD-CA (this chapter)	$\sigma_{\text{CA}} = 9500$, $\sigma^{\mathcal{D}}_{\text{pos}} = 1.5$, $\sigma^{\mathcal{D}}_{\text{vel}} = 25$, $\sigma^{\mathcal{D}}_{\text{acc}} = 1500$	68.3%

‡ For all BD models, $p(\mathcal{T}|\mathcal{D} = i) = \text{Unif}(0.1\text{sec}, 1.9\text{sec})$, the number of quadrature points $N_{\mathcal{T}} = 30$, $\tilde{G} = \mathbf{I}$ and $\sigma^{\mathcal{D}}$ form the corresponding Σ_i.

object. However, such a model might be effective in quickly and accurately predicting the destination.

3. Relying solely on physical meaning may provide an acceptable level of performance. However, when the goal is to edge out competitors by even marginal performance increments, as demonstrated in the upcoming results, such an approach is insufficient. This highlights the importance of the fine-tuning phase or automatic parameter learning techniques. These methods, not strictly limited by physical insights, assist in realising the full potential of each algorithm.

Finally, automatic parameter learning offers a structured alternative to manual tuning. For example, the parameters of OU models may be set based on maximisation of the likelihood $\prod_{k=1}^{K} p(\mathbf{y}^{[k]}_{1:n}|\mathcal{D} = i, \Omega)$ for a sample of K typical full pointing finger trajectories. Here, Ω represents the set of inherent dynamic model parameters (such as the η, ρ, γ in (2.8) on page 26) that are not induced by a specific destination but are common to all N_D destination-driven models. As the driver/passenger uses the touch system, the system can refine the applied model parameters from the larger available dataset(s). However, parameter learning for BD models presents additional challenges, especially with the uncertainty surrounding arrival time. We leave a more formal investigation of parameter choice for future work in these models.

2.5.1 Prediction performance with linear Gaussian intent-driven models

For the 95 pointing tracks covering different levels of perturbations (i.e. Dataset A), the computationally efficient linear Gaussian models are sufficient to predict the intended

icon with a high accuracy. In this section, we evaluate all linear Gaussian models introduced in Section 2.3.1 for this dataset. The parameters for all tested predictors are listed in Table 2.1. They are chosen in a manual way as described above.

The timely successful prediction over pointing duration is shown in Fig. 2.5. As expected, all methods generally exhibit an upward trend, i.e. their performance improves as the the pointing finger-hand approaches the intended endpoint. Specifically, the ERA model can perform poorly at the beginning period of the pointing motion (e.g. in the first 30%); however it deliver comparable results thereafter. Combined with the overall success rate shown in Table 2.1, it can be seen that all examined models achieve comparable prediction successes. Hence, the predictive touch system could infer the intended on-display item remarkably early in the pointing-selection tasks. Nonetheless, it can be noticed from Table 2.1 and Fig. 2.5 that the BD models achieve better results compared with the Gaussian mean reverting models. Although the BD-CV model only exhibits a marginal improvement over the ERA, the BD-CA model shows a clearer superiority compared to Gaussian models, as it outperform all other competing methods almost all the time in Fig. 2.5. Furthermore, performance of models whose acceleration is driven by a Wiener process (BD-CA) are also superior to those constructed merely on target position and velocity (BD-CV). This may be due to the fact that present accelerations can reflect the movement trend with more details. Additionally, the advantage of BD methods may be gained from more accurate end state construction such that a successful prediction can always be achieved at the end of pointing period. It is worth mentioning that in our case the intent predictors implemented according to Algorithm 1 of [70] have the lowest complexity compared with other BD counterparts (please refer to the final paragraph of Section 2.4.1.2 for a comprehensive discussion on the computational complexity of BD methods). Meanwhile, the OU-based predictors have the least computational cost among all evaluated methods.

Note that for better visualisation we have chosen to only display the success rate against gesture time for the BD method proposed in this chapter in Fig. 2.5, because the lines from previous BD formulations are visually very similar to that introduced here. Also, different BD methods in Table 2.1, when the same original Markov model is incorporated, have similar performance. This similarity is expected given that all BD methods share a foundational rationale: the construction of a Markov bridge towards an intended endpoint by integrating knowledge of the destination/terminal state into a standard Markov process. To reiterate, our primary objectives in presenting the BD model transition in Section 2.3.1.2 and introducing its inference procedure in Algorithm

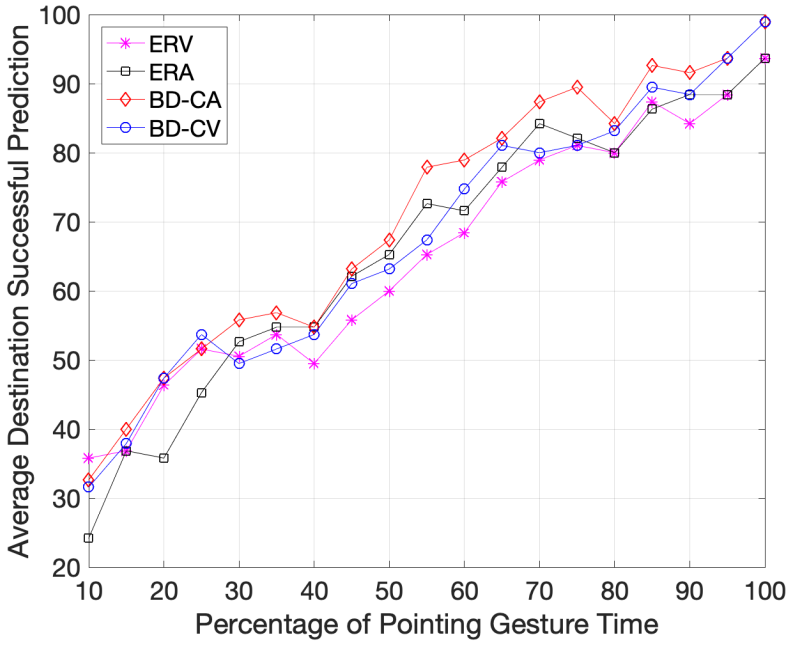


Fig. 2.5 Average successful prediction over time (Dataset A).

1 are not to supersede other BD variants [68, 70] in terms of accuracy or efficiency. Instead, we aim to provide a clear stochastic process interpretation for the BD model and a more intuitive destination prediction methodologies. A detailed elaboration on this perspective is provided in Section 2.1.2. Nevertheless, an important contribution of this chapter is the exploration of the untested BD-CA model. When implemented using either Algorithm 1 or previous BD methods from [68, 70], this model demonstrates superior destination prediction accuracy.

2.5.2 Highly perturbed scenarios and particle filtering

The intent inference performance for highly perturbed trajectories in Dataset B has been tested with jump models and Gaussian mean reverting models in [69] and [86]. Results from the BD models introduced in this chapter are also included for comparison. The aggregate inference success for all algorithms and the timely successful prediction from four selected algorithms (i.e. omitting non-BD models for the clarity of presentation) are depicted in Fig. 2.6 and Fig. 2.7, respectively. The applied jump models below are described in Algorithm 2 and each use 2000 particles, but it has been observed that a comparable performance can be achieved with a small number (e.g. 500) of particles;

Table 2.2 Jump models parameter sets.

Models	Mean-reverting dynamics	Jumps
Jump-ERV	$\eta = 60$, $\rho = 15$, $\sigma = 450$	$\mu_J = 0$, $\sigma_J = 866$, $\lambda_J^{-1} = 1$
Jump-ERA	$\eta = 1150$, $\rho = 320$, $\gamma = 30$, $\sigma = 8000$	$\mu_J = 0$, $\sigma_J = 1.6 \times 10^4$, $\lambda_J^{-1} = 0.2$

their parameters are listed in Table 2.2 (the jumps are assumed to be isotropic) and those for all of the linear Gaussian models remain the same as in Table 2.1.

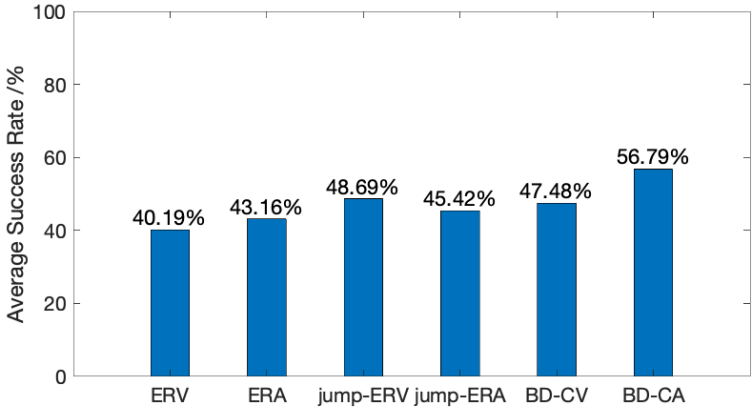


Fig. 2.6 Average success rate for dataset B. ERV was first proposed in [67]. ERA, jump-ERV, and jump-ERA are all introduced in this chapter, with further details available in [69]. BD-CA is implemented and reported here for the first time. While both BD-CV and BD-CA are implemented using Algorithm 1 introduced in this chapter, comparable success rates can be attained using prior BD formulations as presented in [68, 70]. The BD-CA distinctly surpasses other methods, demonstrating a marked improvement in success rate.

From Fig. 2.7, one can see that the BD-CA model always achieves the highest successful prediction after the first 20 percents duration, and the BD models can always achieve the accurate prediction at the end stage of pointing due to its Markov bridge nature. Similar to the linear Gaussian ERA model, the jump-ERA model ascends from a relatively low successful prediction, to a comparable successful rate on the second half of the pointing duration. This insensitivity may be caused by a longer reflection on the observation from the acceleration constructed intention.

The average success rate in Fig. 2.6 indicates that the BD-CA model notably outperforms its counterparts for this dataset, while the jump-ERV model achieves the second best success rate. This superiority of the BD-CA model is further illustrated in Fig. 2.7, where it consistently maintains a robust success rate of 70% starting from just 35% of the overall pointing duration. This suggests that the BD-CA model can

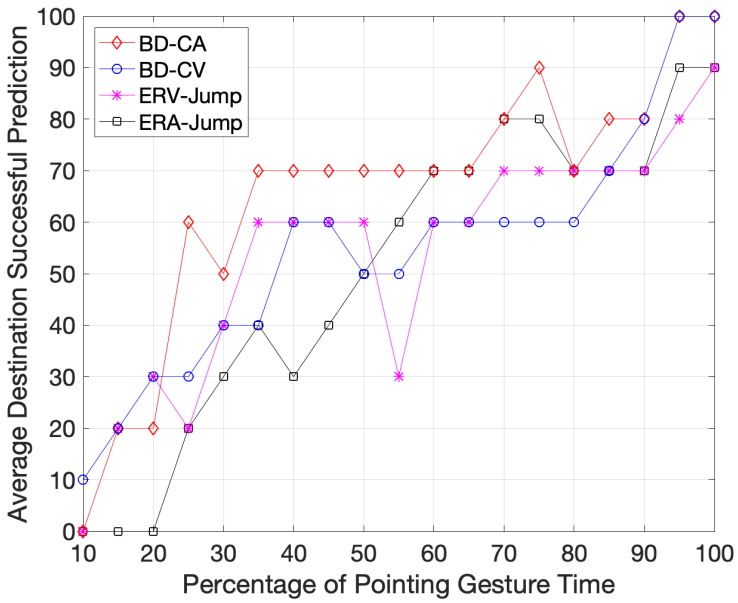


Fig. 2.7 Average successful prediction over time (Dataset B).

facilitate swift and accurate decisions regarding the destination. It's worth noting that while the sub-57% average success rates for all models in Fig. 2.6 might initially appear underwhelming, they are quite good, especially when considering the challenges posed by perturbed pointing data with a vast selection of 37 icons. Importantly, by the 60% mark in the pointing duration — a reasonable juncture to relay predicted destinations for better reliability — all models in Fig. 2.7 archive comparable accuracy levels observed in static-pointing scenarios with only 21 selectable icons, as depicted in Fig. 2.5. This prediction success rate far surpasses a mere random guess, which stands at approximately 2.7%. Additionally, even the least effective model, the ERV, has shown in [67] to outperform more traditional predictors, like the nearest-neighbor-based methods.

The results illustrated above may lead to the conclusion that the BD-CA is the best among other models on characterising the intention of the hand pointing. However, it is worthwhile to note that the exploration for the parameters of jump models are more restrictive due to their larger number of parameters and time-consuming evaluation process. Thus it is possible that a better results can be achieved with other parameters for jump models, especially for the jump-ERA model. Additionally, the present jumps/jolts in those 10 tracks might not be of the severity (magnitude and/or transience) that a BD-CA model cannot successfully smooth out or follow. Under such high-levels of perturbations, the numerical marginalisation of arrival time with BDs can

be challenging as the pointing-task duration can be subject to large delays, with the risk of it being very distinctive from the prior of $\mathcal{T}$. Nevertheless, the use of the particle filtering with a jump process offers additional advantages, not necessarily relevant to the predictive touch usecase, such as detecting the location-time of the perturbations-induced fast manoeuvres (jumps) and potentially better destination-aware tracking results, see [69].

2.6 Conclusion

In this chapter, we presented an overview of the existing stochastic dynamic modelling methods for destination inference, with the in-vehicle predictive touch system as the case study. It covers linear Gaussian and nonlinear setups, both proposed within a Bayesian framework. The adopted continuous time intent-driven state space models naturally facilitate treating asynchronous data, including from multiple sensors. In addition, a new bridging distribution approach was proposed here, which has a moderate computational requirement and a clear stochastic interpretation compared with previous formulations. Results from real data of a predictive touch system demonstrated the efficacy of the various considered prediction algorithms, namely their ability to infer the user intent remarkably early in the pointing-selection task. Thereby, this can facilitate effective touchless interactions via the intuitive free hand pointing gestures. It is emphasised that the presented prediction techniques are also applicable to other fields, e.g. surveillance, smart navigation, robotics, etc. Nevertheless, there are several extensions to this work, for example bridging distributions for non-linear and/or non-Gaussian systems (e.g. a stable Lévy system in [91]), considering intrinsically nonlinear intent-driven motion models for highly manoeuvrable objects and various measurement models (one such example can be found in [92]). This chapter serves as an impetus to further research on meta-level tracking models and inference algorithms.

Chapter 3

Lévy State-space Models for Tracking and Intent Prediction of Highly Manoeuvrable Objects

Typically the dynamic models in a tracking application are devised as a random process in state-space form [10], of which the most commonly used is the linear Gaussian model (e.g. [93, 10, 68] and many models introduced in previous chapters) owing to its analytical tractability. However, a Gaussian model assumes a normally distributed state transition between any time interval, which renders it inappropriate for modelling of the occasional abrupt changes of the state exhibited during sharp manoeuvring behaviour. To better characterise such erratic manoeuvring behaviour, this chapter presents α-stable Lévy state-space models, expressed in continuous time as Lévy processes. In contrast to conventional (fully) Gaussian formulations, the proposed models are driven by heavy-tailed α-stable noise and are thus much more able to capture extreme values/behaviours. The model is designed here for both object tracking and intent inference applications.

In particular, the model is represented in a conditionally Gaussian series form which ensures the tractability of the applied inference algorithms. A corresponding estimation strategy with the Rao-Blackwellised particle filter is then proposed and an efficient intent inference procedure is introduced. Here, the underlying intent, driving the target's long-term behaviour (e.g. reaching its final destination), is modelled as a latent variable. Real vessel data from maritime surveillance and human computer interactions (e.g. cursor data from motor-impaired interface users) are utilised to demonstrate the effectiveness of the proposed approach. It is shown to deliver noticeable improvements in the tracking and intent prediction performance (whenever relevant) compared with

a more conventional Gaussian dynamic model. This chapter includes results that have been previously published in [94][1][95][2].

3.1 Introduction

As demonstrated in Chapter 2, there has been a growing interest in the object tracking field in inferring, as rapidly as possible, a target's intent (e.g. its final destination and future motion patterns) from the available sensor measurements. This aim is motivated by numerous applications, such as surveillance, sensor management, automation and robotics. Intent prediction can be considered as a *meta-level* tracking task [6], whose objective is to acquire a higher-level understanding of a scene by revealing the underlying intent driving the target's long-term behaviour (e.g. to reach an endpoint). This is in contrast with conventional *sensor-level* trackers [4, 96], aimed at estimating solely the object temporal kinematics (e.g. position, velocity, heading, etc.).

Building upon and extending the Bayesian framework for intent prediction presented in Chapter 2, in this chapter, we address the problem of estimating both the kinematic state and intent (typically the final destination), especially for highly manoeuvrable objects using the stable Lévy model. Whilst the unknown endpoint cannot be directly observed, stochastic models can be devised that capture the influence of intent on the evolution of the observed target state over time. Constructing representative dynamic models for fast manoeuvring objects is generally challenging, especially if intent-driven, since a target trajectory can feature large accelerations and rapid rotational motions, for example by vehicles for obstacle avoidance, drones, vessels [88], insects undertaking erratic movements [97, 98], cursor pointing motion of users with severe motor impairments [90], to list a few. Here, a novel modelling approach for manoeuvring targets that can incorporate both kinematics and intent is proposed. The methods are a novel extension of continuous-time Lévy state-space models [18, 21], driven by heavy-tailed α-stable Lévy processes [99, 100]. These can better characterise sudden dramatic changes in the state induced by swift target manoeuvres, compared with the more commonly used Gaussian-driven dynamic model.

If the dynamic noise in a system can be considered as the combination of a large number independent identically distributed perturbations, it is reasonable to model it using an α-stable distribution. The generalised central limit theorem [101] supports this choice, as it suggests that the sum of such perturbations converges under appropriate

[1] © 2021 IEEE. Reprinted, with permission, from [94]
[2] © 2020 IEEE. Reprinted, with permission, from [95]

scaling to a member of the α-stable distribution class. The conventional Gaussian noise is a special case within this class, and thus the α-stable noise in the proposed Lévy state-space model serves as a natural and more general extension to handle heavy-tailed cases. Furthermore, the parameterised skewness and heavy-tailedness of the α-stable noise offer flexibility for characterising a wide range of system noise [102, 103]. In the continuous time setup, such an α-stable noise can be considered as the derivative of an α-stable Lévy process, which shares several useful properties (e.g. self-similarity and infinite divisibility) with the prevalent Brownian motion. Consequently, the α-stable process has been studied in various areas (e.g. telecommunications, econometrics, and signal processing [104–106]) to model extreme events or abrupt changes.

3.1.1 Contributions

The main contribution of this chapter is introducing for the first time a stable Lévy modelling approach for multidimensional spatial tracking and intent prediction, both of which are addressed using a Bayesian framework similar to that in Chapter 2. Detailed derivations of this novel generic formulation, with parameterised α-stable dynamic noise, are provided. We also show how it can be applied to estimate the kinematic state and/or destination of a manoeuvring target, presenting several different example models and inference strategies. Notably, our introduced stable Lévy modelling and inference framework is highly adaptable, routinely enabling the integration of numerous established continuous-time dynamic models by substituting traditional Gaussian-driven noise with α-stable noise, thereby improving manoeuvrability.

From a theoretical perspective, the proposed stable Lévy model is an innovative extension of the continuous-time Lévy state-space models [21]. While the model in previous work is driven by a single scalar heavy-tailed α-stable Lévy process, ours is distinguished by its capability to be driven by multiple isotropic multivariate α-stable Lévy processes, a feature that makes it particularly suited for spatial tracking applications. Our model can incorporate the unknown intent of a tracked object using a mean-reverting term. Furthermore, it is devised with a conditionally Gaussian infinite series, which extends the series for scalar α-stable processes as described in [21, 107, 100]. While the conditionally Gaussian series structure is theoretically exact, but in practical applications requires the truncation of the infinite series and suitable approximation of the residual error terms. Owing to the employed conditionally Gaussian structure, both of the manoeuvring target kinematic state and intent can be inferred with an efficient Rao-Blackwellised particle filter [45, 49].

Additionally, we propose a latent destination model, where an extended latent state that incorporates destination is constructed. The intent prediction task with such a model then becomes a by-product of the state estimation (tracking) algorithm, thereby requiring minimal additional computations. It also allows a dynamically changing or static intended endpoint, which can be anywhere within the surveyed area and does not have to be confined to a predefined finite set of possible destinations, unlike prior work, e.g. in [68] and intent prediction methods introduced in Chapter 2.

Results from real pointing and maritime surveillance data are presented to demonstrate that the competitive performance of the proposed modelling and inference approach, in terms of determining the kinematic state and final destination (when relevant) of a target whose trajectory can exhibit sharp manoeuvres. The utilised pointing data pertains to users with severe motor-impairment (namely cerebral palsy) pointing in 2-D on an interface with a mouse cursor and participants in a moving vehicle undertaking freehand pointing gestures in 3-D to select icons on the in-car display whilst experiencing severe perturbations due to the road/driving conditions (i.e. vibrations and accelerations). It is emphasised that the proposed stable Lévy models are also readily applicable for manoeuvring object tracking, i.e. without intent prediction, as in the reported maritime vessel tracking example in Section 3.6.3.

3.1.2 Outline

The remainder of the chapter is organised as follows: We first list the contributions in this chapter before reviewing several related works. The proposed stable Lévy modelling framework is described in Section 3.2 where exact and truncated series representations of the α-stable Lévy integral are provided. Section 3.3 then details the multidimensional state-space form of the stable Lévy model, with example models in Section 3.4. The inference algorithms for estimating the target state and intent are defined in Section 3.5. Results from real data of various scenarios are shown in Section 3.6 and conclusions are drawn in Section 3.7.

3.1.3 Related work

Whilst the widely adopted linear Gaussian state-space model with additive Gaussian noise [10] for sensor or meta-level tracking produces simple closed form inference methods [93, 73, 68], they cannot accurately characterise abrupt changes in the target (kinematic) state caused by sudden manoeuvres. The Gaussian premise stipulates that such drastic changes are very unlikely, producing large estimation errors when they

occur. More sophisticated models, which include a jump process in addition to the original Gaussian process such as the jump diffusion models or Markov/semi-Markov jump models, show improved inference results for manoeuvring targets [108, 69]. In contrast, we utilise in this chapter an α-stable Lévy process as the driving noise, which consists of fewer parameters and leads to competitive inference performance. Other manoeuvring models, e.g. the constant turn [4] and intrinsic coordinates [88] models, rely on the object's speed and/or heading information. Whilst these are more fitting for curvilinear motion modelling, the model introduced in this chapter is formulated in Cartesian coordinates with the aim of straightforward application to diverse tracking scenarios.

Despite their ability to capture highly manoeuvring behaviour through extreme changes in latent state values, stable Lévy models are usually highly intractable for inference due to the lack of a closed-form density function for the increments of the process. This has resulted in their limited use in the object tracking field, though see [109] for a discrete time tracking model that uses stable law innovations to achieve modelling of fast manoeuvres. To overcome this, a conditionally Gaussian Poisson series representation was introduced in [110–113] to represent the α-stable random variable and its stochastic integral, which can form the basis of a tractable inference framework for the stable Lévy process family [113, 107]. Sharp rates of convergence for these approaches have been recently proposed, see [114] and references therein. A more general Lévy state-space model was detailed in [21], however not as a spatial process applicable to object tracking. Here, we propose for the first time multidimensional Lévy state-space models in the conditionally Gaussian series form for spatial tracking and intent prediction, coupled with an efficient sequential Bayesian inference procedure based on Rao-Blackwellised particle filtering. On account of the typically isotropic properties in most tracking applications, we focus particularly on the symmetric stable Lévy process. Although skewed dynamic models may be appropriate in certain scenarios and can be included in our framework as in [21], we leave this as a topic for future work.

As a further motivation for our work, there is a growing interest in the application of non-Gaussian Lévy processes for animal behaviour modelling such as the 'Lévy flight' and 'Lévy walk' models [115]. In these approaches the searching or foraging patterns of various species (e.g. fruit fly, albatross, bumblebee, etc.) can be approximated as a heavy-tailed Lévy process, where occasional long distance steps in foraging animal trajectories correspond to the extreme noise in the Lévy process, see [116–118]. Nonetheless, this is often applied only informally for (hypothesis) validation purposes

and not within a rigorous tracking formulation. For example, the 'Lévy flight' and 'Lévy walk' models in the biology field include only spatial position as a random variable, and this is usually time-discretised for ease of implementation. This renders them inappropriate for tracking with frequent (asynchronous) observations, in which the estimation of higher order kinematics such as velocity can provide valuable information, enhance accuracy, and potentially lead to deeper understanding of animal dynamics and the hypotheses that may apply. We note that our proposed modelling and inference frameworks can be readily applied to these animal tracking scenarios, including agile animals, such as fruit flies or bumblebees [97, 98]. In particular, fruit flies can change their direction by roughly 90 degrees in just about 50 ms [98]. This tracking task will be studied in future work.

Finally, model-based intent predictors can be decoupled from the sensor-level tracker for computational efficiency or due to system requirements, e.g. legacy algorithms or system architectures (see [70, 6]). Alternatively, the intent information can be directly incorporated into the target dynamic model, with potentially simplified meta-level tracker and more accurate target state estimation. If determining the object endpoint is sought, the latter intent-driven models are usually destination-reverting such as the linear Gaussian models based on *Ornstein-Uhlenbeck* process [73, 67], bridging distributions [68, 70] or conditionally Markov process/reciprocal process [76, 75, 6], and jump diffusion models [69]. Another modelling approach, which can also detect anomalous trajectories or rendezvousing, employs context-free grammars [6, 75, 119]. It discretises the state space where the target movements are assumed to be constrained (e.g. confined to a road map). Here, we tackle the intent inference task for manoeuvring objects or perturbed trajectories with stable Lévy state space models, capitalising on their heavy-tailed property to capture abrupt changes in the target motion, involving no discretisation of the state space or constraints on the object path. We also present a more computationally efficient inference strategy compared with [67–70] by modelling intent within an extended dynamical state that is inferred as part of the tracking algorithm. This allows for a fixed computational cost, unlike previous methods where the cost increased linearly with the number of potential destinations. With the introduced latent destination models, the intended endpoint can be any spatial point/region within a surveyed area, i.e. not necessarily one of a finite set of predefined destinations as in [67–70, 91].

3.2 Stable Lévy modelling framework

We define a latent dynamical state in the spatial domain,

$$\boldsymbol{x}(t) = [\boldsymbol{x}_1(t)^\top, \boldsymbol{x}_2(t)^\top, ..., \boldsymbol{x}_m(t)^\top]^\top, \tag{3.1}$$

where $\boldsymbol{x}_j(t) \in \mathbb{R}^{s_j}$, $j = 1, 2, ..., m$, is the vector of the s_j hidden variables (e.g. position and velocity of an object) in the j^{th} spatial dimension, and m is the number of considered dimensions (e.g. $m = 2$ for planar and $m = 3$ for 3-D motion). Specific forms for $\boldsymbol{x}_j(t)$ are given in Section 3.4. The stable Lévy model is formulated as a continuous-time linear system driven by multivariate isotropic α-stable noise.

For the canonical case, where there is only one multivariate isotropic noise on $\mathbb{R}^m$, the proposed stable Lévy model can be described by the following stochastic differential equation (SDE),

$$d\boldsymbol{x}(t) = \boldsymbol{A}\boldsymbol{x}(t)dt + \boldsymbol{B}dt + \boldsymbol{H}d\boldsymbol{W}(t), \tag{3.2}$$

where $\boldsymbol{B}$ is a known input vector, e.g. serves as input in a mean reverting model [73, 68, 69], and,

$$\boldsymbol{A} = diag(\boldsymbol{a}_1, ..., \boldsymbol{a}_m), \quad \boldsymbol{B} = [\boldsymbol{b}_1^\top, \boldsymbol{b}_2^\top, ..., \boldsymbol{b}_m^\top]^\top, \quad \boldsymbol{H} = \begin{bmatrix} \boldsymbol{h}_1 & \boldsymbol{0} & \dots & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{h}_2 & \dots & \boldsymbol{0} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{0} & \boldsymbol{0} & \dots & \boldsymbol{h}_m \end{bmatrix}, \tag{3.3}$$

such that $\boldsymbol{a}_j \in \mathbb{R}^{s_j \times s_j}$, $\boldsymbol{b}_j \in \mathbb{R}^{s_j}$ and $\boldsymbol{h}_j \in \mathbb{R}^{s_j}$ characterise the transition function for the j^{th} dimension. $\boldsymbol{W}(t)$ is the isotropic multivariate α-stable Lévy process on $\mathbb{R}^m$ [100] which implies:

1. $\boldsymbol{W}(0) = \boldsymbol{0}$ almost surely[3],

2. Independent, isotropic stationary increments $\boldsymbol{W}(t) - \boldsymbol{W}(s)$, $t > s$, having characteristic function

$$\phi(\boldsymbol{\theta}) = \mathrm{E}[e^{i\boldsymbol{\theta}(\boldsymbol{W}(t)-\boldsymbol{W}(s))}] = e^{-\sigma^\alpha(t-s)|\boldsymbol{\theta}|^\alpha}, \tag{3.4}$$

[3]In probability theory, an event is said to happen almost surely if it happens with probability 1 (or Lebesgue measure 1).

where α is the stability parameter on $(0, 2]$ and σ is the scale parameter. A special case is when $\alpha = 2$, and then $\boldsymbol{W}(t)$ is the m-dimensional isotropic Brownian motion. We focus in this paper on stable Lévy models with isotropic noise, as is common in much of the tracking literature [10, 96, 4]; however we note that the more general skewed case in [21] could also be extended to the current multidimensional setting in future work. The isotropic premise supports the typical lack of information about directionality of the randomness of dynamics in a tracking scenario.

The solution of the SDE in (3.2) can be written as [100]

$$\boldsymbol{x}(t + \Delta t) = e^{\boldsymbol{A}\Delta t}\boldsymbol{x}(t) + \int_0^{\Delta t} e^{\boldsymbol{A}(\Delta t - u)}\boldsymbol{B}du + \int_0^{\Delta t} e^{\boldsymbol{A}(\Delta t - u)}\boldsymbol{H}d\boldsymbol{W}(u). \tag{3.5}$$

The first two terms are familiar from the solution of standard Brownian-driven models. The third term is a stable stochastic integral and requires special attention, see later development. The block-diagonal structure of the model matrices $\boldsymbol{H}$ and $\boldsymbol{A}$ in (3.3) is familiar from the structure of standard coordinate-uncoupled Gaussian models [10]. In the stable case however, the coordinates become coupled through certain latent random variables and are thus not treated independently of one another.

A more general stable Lévy system may be driven by multiple independent isotropic noises, i.e.

$$d\boldsymbol{x}(t) = \boldsymbol{A}\boldsymbol{x}(t)dt + \boldsymbol{B}dt + \sum_{k=1}^{N_{\mathcal{W}}} \boldsymbol{H}_k d\boldsymbol{W}_k(t), \tag{3.6}$$

where the $\boldsymbol{W}_k(t)$, $k = 1, 2, ..., N_{\mathcal{W}}$, are isotropic α-stable processes independent of each other and $N_{\mathcal{W}}$ is the number of such processes; $\boldsymbol{W}_k(t)$ can be of arbitrary dimension and will be mapped to the desired coordinate by $\boldsymbol{H}_k$. Such a SDE can describe the dynamics in more diverse scenarios, such as multiple interacting targets or single target with a linear combination of noise sources. The block definition for $\boldsymbol{x}(t), \boldsymbol{A}$ and $\boldsymbol{B}$ remain the same and their specific elements should be defined as required. e.g. for multi-targets scenario, $\boldsymbol{x_j}$ should include kinematics from different objects, $\boldsymbol{a_j}$ should specify their interactions, and each $\boldsymbol{W}_k(t)$ for each object. A specific example of such a system will be given in Section 3.3.2. Similar to (3.5), the solution of this SDE is

$$\boldsymbol{x}(t + \Delta t) = e^{\boldsymbol{A}\Delta t}\boldsymbol{x}(t) + \int_0^{\Delta t} e^{\boldsymbol{A}(\Delta t - u)}\boldsymbol{B}du + \sum_{k=1}^{N_{\mathcal{W}}} \int_0^{\Delta t} e^{\boldsymbol{A}(\Delta t - u)}\boldsymbol{H}_k d\boldsymbol{W}_k(u). \tag{3.7}$$

It can be noted from (3.5) and (3.7) that the only term that cannot be calculated explicitly in each case is the final α-stable stochastic integral. Without loss of generality,

we consider this integral with respect to a $\boldsymbol{W}(t)$ on $\mathbb{R}^m$ (it can be a $\boldsymbol{W}_k(t)$ in (3.7) with m of the appropriate size), and correspondingly define

$$\boldsymbol{I}(\Delta t) = \int_0^{\Delta t} \mathbf{f}(\Delta t, u) d\boldsymbol{W}(u), \tag{3.8}$$

with

$$\mathbf{f}(\Delta t, u) = e^{\boldsymbol{A}(\Delta t - u)} \boldsymbol{H}.$$

We next introduce the following two representations for this intractable integral: A) an exact Poisson series representation, which is in terms of an infinite series, and B) a truncated series representation, which approximates the exact representation accurately and is then utilised to devise tractable stable Lévy models in Sections 3.3 and 3.4.

3.2.1 Exact series representation of the α-stable Lévy integral

With a direct multivariate extension of the Poisson series representation in [100, 21], the stochastic integral in (3.8) can be exactly expressed as

$$\boldsymbol{I}(\Delta t) \overset{D}{=} \Delta t^{1/\alpha} \sum_{i=1}^{\infty} \Gamma_i^{-1/\alpha} \mathbf{f}(\Delta t, V_i) \boldsymbol{U}_i, \tag{3.9}$$

where $\overset{D}{=}$ denotes both sides having the same distribution,

- $\{\Gamma_i\}$ are the unit rate Poisson process arrival times (epochs), i.e. $(\Gamma_{i+1} - \Gamma_i) \sim$ exp(1) and exp(1) is an exponential distribution with mean 1;
- $\{V_i\}$ are independent and identically distributed (i.i.d.) random variables uniformly distributed in $(0, \Delta t]$, and
- $\{\boldsymbol{U}_i\}$ are any i.i.d. isotropic random vectors with finite α-th moment. Here, we choose to use $\boldsymbol{U}_i \sim \mathcal{N}(\mathbf{0}, \sigma_W^2 I_m)$, where $I_m \in \mathbb{R}^{m \times m}$ is the identity matrix, to construct a conditionally Gaussian form of $\boldsymbol{I}(\Delta t)$ for tractable inference.

In a typical single object tracking setup where $\boldsymbol{a}_j, \boldsymbol{h}_j$ in (3.3) are constants across different dimensions j, this series (3.9) represents an isotropic random variable. Nonetheless, the non-isotropic case can be easily included through a non-zero mean term and non-diagonal covariance matrix for $\{\boldsymbol{U}_i\}$, For discussion of this case, including certain required compensation terms, see [21] and references therein.

3.2.2 Truncated series representation

Note that the above series representation, although exact in terms of the distribution of $\boldsymbol{I}(\Delta t)$, has an infinite number of terms. For practical implementation we introduce a truncated series representation to approximate (3.9). Observe from (3.9) that the Poisson terms $\{\Gamma_i^{-1/\alpha}\}_{i=1}^{\infty}$ are strictly decreasing with i, and hence a reasonable approach is to perform a random truncation of the series at some value of i. Noting that the scale of series increases with Δt, we propose the truncation $\{i : \Gamma_i < c\Delta t\}$, $c \in (0, \infty)$, as in [21], and the expected number of terms in the truncated series thus grows linearly with the time interval Δt. Accordingly, the truncated series can be expressed as

$$\boldsymbol{I}_c(\Delta t) = \Delta t^{1/\alpha} \sum_{\Gamma_i < c\Delta t} \Gamma_i^{-1/\alpha} \mathbf{f}(\Delta t, V_i) \boldsymbol{U}_i. \tag{3.10}$$

To compensate for the error due to the series truncation in (3.10), we introduce an approximation to the resulting residual error series. Such a residual series can be expressed as

$$\begin{aligned} \boldsymbol{R}_c(\Delta t) =& \boldsymbol{I}(\Delta t) - \boldsymbol{I}_c(\Delta t) \\ =& \Delta t^{1/\alpha} \sum_{\Gamma_i \geq c\Delta t} \Gamma_i^{-1/\alpha} \mathbf{f}(\Delta t, V_i) \boldsymbol{U}_i. \end{aligned} \tag{3.11}$$

To characterise this residual series, we compute its first and second moments. Since $\{\boldsymbol{U}_i\}$ have zero mean and are independent from the other variables, its mean is given by

$$\begin{aligned} \mathrm{E}[\boldsymbol{R_c}(\Delta t)] =& \mathrm{E}[\mathrm{E}_{U_i}[\boldsymbol{R_c}(\Delta t)]] \\ =& \mathrm{E}[\Delta t^{1/\alpha} \sum_{\Gamma_i \geq c\Delta t} \Gamma_i^{-1/\alpha} \mathbf{f}(\Delta t, V_i) \mathrm{E}_{U_i}[\boldsymbol{U}_i]] = \mathbf{0}. \end{aligned}$$

The detailed derivation of the variance of the residual terms is in Appendix 3.A. This derivation follows largely from Section 8.2.1 of [120], with a notable adjustment being the use of our multivariate variables $\boldsymbol{U}_i$ instead of scalar variables.

Specifically, we have

$$\mathrm{Var}[\boldsymbol{R_c}(\Delta t)] = \sigma_W^2 \frac{\alpha}{2-\alpha} c^{1-\frac{2}{\alpha}} \boldsymbol{Q}(\Delta t),$$

$$\boldsymbol{Q}(\Delta t) = \int_0^{\Delta t} e^{\boldsymbol{A}(\Delta t - u)} \boldsymbol{H} \boldsymbol{H}^\top e^{\boldsymbol{A}(\Delta t - u)^\top} du. \tag{3.12}$$

Consequently, we can approximate this residual series with a Gaussian variable which exactly matches the first and second moments of $\boldsymbol{R}_c(\Delta t)$, similarly to the procedure in [21], i.e.

$$\tilde{\boldsymbol{R}}_c(\Delta t) \sim \mathcal{N}\left(\boldsymbol{0}, \sigma_W^2 \frac{\alpha}{2-\alpha} c^{1-\frac{2}{\alpha}} \boldsymbol{Q}(\Delta t)\right), \tag{3.13}$$

The fully truncated version for the isotropic stable Lévy integral is then conditionally Gaussian and expressed as

$$\boldsymbol{I}(\Delta t) \approx \boldsymbol{I}_c(\Delta t) + \tilde{\boldsymbol{R}}_c(\Delta t). \tag{3.14}$$

This accurate Gaussian approximation can be justified for large c by central limit theorem on $\boldsymbol{R}_c(\Delta t)$, see [21, 121, 114] and in particular the sharp convergence bounds for the symmetric scalar case reported in [114].

3.3 Stable Lévy state space model

The state-space form of the stable Lévy model can now be developed based on the truncated Poisson series representation in (3.14). For notational convenience, we define

$$\begin{aligned} \boldsymbol{F}(\Delta t) &= e^{\boldsymbol{A}\Delta t}, \\ \boldsymbol{M}(\Delta t) &= \int_0^{\Delta t} e^{\boldsymbol{A}(\Delta t - u)} \boldsymbol{B} du. \end{aligned} \tag{3.15}$$

Recall that our stable Lévy model can be driven by one or multiple α-stable noise(s) as specified in the SDEs in (3.2) and (3.6). Here, we first formulate the state-space form of the canonical single noise case described by (3.2). The multiple noises setting in (3.6) can then be similarly obtained by simply summing the multiple truncated series representations. However, in the example presented in this chapter we will use a special-case multiple noises scenario where it consists of a non-Gaussian α-stable noise and a Gaussian noise, both on $\mathbb{R}^m$.

3.3.1 Model driven by a single α-stable noise source

The stable Lévy model with a single α-stable noise has the SDE solution in (3.5). Substituting the series representations in (3.14) into (3.5) yields

$$\begin{aligned}
&\boldsymbol{x}(t+\Delta t)\\
=&e^{\boldsymbol{A}\Delta t}\boldsymbol{x}(t)+\int_0^{\Delta t}e^{\boldsymbol{A}(\Delta t-u)}\boldsymbol{B}du+\int_0^{\Delta t}e^{\boldsymbol{A}(\Delta t-u)}\boldsymbol{H}d\boldsymbol{W}(u)\\
\approx&\boldsymbol{F}(\Delta t)\boldsymbol{x}(t)+\boldsymbol{M}(\Delta t)+\boldsymbol{I}_c(\Delta t)+\tilde{\boldsymbol{R}}_c(\Delta t).
\end{aligned}\tag{3.16}$$

Note that given all the required $\{\Gamma_i\}_{(0,c\Delta t]}$ and $\{V_i\}_{(0,\Delta t]}$, $\boldsymbol{I}_c(\Delta t)$ in (3.10) is the summation of independent Gaussian vectors with zero mean and covariance being $\Delta t^{2/\alpha}\Gamma_i^{-2/\alpha}\sigma_W^2\mathbf{f}(\Delta t,V_i)\mathbf{f}(\Delta t,V_i)^\top$, and thus $\boldsymbol{I}_c(\Delta t)$ follows a conditionally normal distribution, that is

$$p(\boldsymbol{I}_c(\Delta t)|\{\Gamma_i,V_i\}_{\Delta t}^c)=\mathcal{N}\left(\mathbf{0},\Delta t^{2/\alpha}\sigma_W^2\sum_{\Gamma_i<c\Delta t}\Gamma_i^{-2/\alpha}\mathbf{f}(\Delta t,V_i)\mathbf{f}(\Delta t,V_i)^\top\right),\tag{3.17}$$

where we define

$$\{\Gamma_i,V_i\}_{\Delta t}^c=\{\{\Gamma_i\}_{(0,c\Delta t]},\{V_i\}_{(0,\Delta t]}\}.$$

Since the residual approximation $\tilde{\boldsymbol{R}}_c(\Delta t)$ is another independent Gaussian vector as given in (3.13), the transition density for the α-stable Lévy state-space model can be computed according to (3.16) as

$$p(\boldsymbol{x}(t+\Delta t)|\boldsymbol{x}(t),\{\Gamma_i,V_i\}_{\Delta t}^c)=\mathcal{N}(\boldsymbol{F}(\Delta t)\boldsymbol{x}(t)+\boldsymbol{M}(\Delta t),\boldsymbol{S}(\Delta t)),$$

where

$$\boldsymbol{S}(\Delta t)=\Delta t^{2/\alpha}\sigma_W^2\sum_{\Gamma_i<c\Delta t}\Gamma_i^{-2/\alpha}\mathbf{f}(\Delta t,V_i)\mathbf{f}(\Delta t,V_i)^\top+\sigma_W^2\frac{\alpha}{2-\alpha}c^{1-\frac{2}{\alpha}}\boldsymbol{Q}(\Delta t).\tag{3.18}$$

The mean of this transition density, i.e. $\boldsymbol{F}(\Delta t)\boldsymbol{x}(t)+\boldsymbol{M}(\Delta t)$, reflects the overall motion trend of the target. By adjusting the parameters $\boldsymbol{A}$ and $\boldsymbol{B}$ in the SDE in (3.2), this model covers a wide range of linear dynamics for diverse applications, e.g. mean-reverting for intent inference or a constant velocity/acceleration behaviour for tracking. The noise parameters α and σ_W characterise the manoeuvrability (small α for more heavy-tailed) and scaling, respectively. As this model is driven by a single α-stable noise, it is suitable for a single manoeuvring object, for both tracking and intent inference applications. Examples can be found in Section 3.4.

3.3.2 Model driven by both non-Gaussian α-stable and Gaussian noises

The focus here is on multiple α-stable noises in (3.6), in particular two noises on $\mathbb{R}^m$ such that one is with $0 < \alpha < 2$ and the other has $\alpha = 2$ (i.e. Gaussian). Since the Gaussian stochastic integral has well-known closed-form solution, we only employ the Poisson series representations for the non-Gaussian integral. Specifically, the SDE in (3.6) reduces to

$$d\boldsymbol{x}(t) = \boldsymbol{A}\boldsymbol{x}(t)dt + \boldsymbol{B}dt + \boldsymbol{H}d\boldsymbol{W}(t) + \boldsymbol{G}d\boldsymbol{\beta}(t), \tag{3.19}$$

where $\boldsymbol{\beta}(t)$ is the isotropic Brownian motion on $\mathbb{R}^m$, with the covariance matrix $\sigma_B^2 I_m$; $\boldsymbol{G}$ has a block-diagonal structure similar to $\boldsymbol{H}$, i.e.

$$\boldsymbol{G} = \begin{bmatrix} \boldsymbol{g}_1 & \boldsymbol{0} & \dots & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{g}_2 & \dots & \boldsymbol{0} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{0} & \boldsymbol{0} & \dots & \boldsymbol{g}_m \end{bmatrix}, \tag{3.20}$$

such that $\boldsymbol{g}_j \in \mathbb{R}^{s_j}$, with $j = 1, 2, ..., m$, maps the j-th scalar components of $\boldsymbol{\beta}(t)$ to the desired latent variables in the j-th dimension. The stable Lévy state-space model for this setup is

$$\boldsymbol{x}(t + \Delta t) \approx \boldsymbol{F}(\Delta t)\boldsymbol{x}(t) + \boldsymbol{M}(\Delta t) + \boldsymbol{I}_c(\Delta t) + \tilde{\boldsymbol{R}}_c(\Delta t) + \int_0^{\Delta t} e^{\boldsymbol{A}(\Delta t - u)}\boldsymbol{G}d\boldsymbol{\beta}(u). \tag{3.21}$$

We still have the conditionally Gaussian form of $\boldsymbol{I}_c(\Delta t)$ as (3.17), and the following transition density can thus be attained

$$p(\boldsymbol{x}(t + \Delta t)|\boldsymbol{x}(t), \{\Gamma_i, V_i\}_{\Delta t}^c) = \mathcal{N}(\boldsymbol{F}(\Delta t)\boldsymbol{x}(t) + \boldsymbol{M}(\Delta t), \boldsymbol{S}(\Delta t)),$$

where

$$\begin{aligned} \boldsymbol{S}(\Delta t) =& \Delta t^{2/\alpha}\sigma_W^2 \sum_{\Gamma_i < c\Delta t} \Gamma_i^{-2/\alpha} \mathbf{f}(\Delta t, V_i)\mathbf{f}(\Delta t, V_i)^\top \\ &+ \sigma_W^2 \frac{\alpha}{2 - \alpha} c^{1 - \frac{2}{\alpha}} \boldsymbol{Q}(\Delta t) + \sigma_B^2 \boldsymbol{\Omega}(\Delta t), \end{aligned} \tag{3.22}$$

$$\boldsymbol{\Omega}(\Delta t) = \int_0^{\Delta t} e^{\boldsymbol{A}(\Delta t - u)}\boldsymbol{G}\boldsymbol{G}^\top e^{\boldsymbol{A}(\Delta t - u)^\top} du. \tag{3.23}$$

A special case for this model is when $\sigma_B = 0$, i.e. the covariance matrix for Brownian motion $\boldsymbol{\beta}(t)$ is zero or negligible, the model reduces to the single α-stable noise driven case in the Section 3.3.1. Typically, this additional Brownian motion aims to provide Gaussian randomness for multiple target dynamics modelling. Alternatively, it can, as well as the α-stable noise, apply to the same object. One example of this model for intent inference application is detailed in Section 3.4.2.

3.4 Model examples

In this section, we give concrete examples of models within the stable Lévy state space framework as defined in Section 3.3. Due to the large number of models considered below, we do not provide their explicit discrete form, instead we introduce the general schemes to compute the required terms $\boldsymbol{M}(\Delta t), \boldsymbol{Q}(\Delta t),$ and $\boldsymbol{\Omega}(\Delta t)$. These schemes are based on the assumption that the matrix exponential can be computed easily and accurately; hence by such an assumption $\boldsymbol{F}(\Delta t)$ in (3.15) is already easily calculated.

Let the square matrix $\boldsymbol{A}$ have dimension $s_A \times s_A$, i.e. $s_A = \sum_{j=1}^{m} s_j$. We start with the simpler integral $\boldsymbol{M}(\Delta t)$ in (3.15). It can be noticed that if $\boldsymbol{A}$ is invertible, we have a straightforward expression

$$\boldsymbol{M}(\Delta t) = \boldsymbol{A}^{-1}(e^{\boldsymbol{A}\Delta t} - I_{s_A})\boldsymbol{B}. \tag{3.24}$$

Here, I propose a method to compute $\boldsymbol{M}(\Delta t)$ regardless of the structure of $\boldsymbol{A}$ (including the singular case) such that

$$\begin{aligned} \boldsymbol{M}(\Delta t) =& \boldsymbol{J}_M \boldsymbol{B}, \\ \begin{bmatrix} \boldsymbol{J}_M \\ I_{s_A} \end{bmatrix} =& \text{expm}\left(\begin{bmatrix} \boldsymbol{A} & I_{s_A} \\ \boldsymbol{0}_{s_A} & \boldsymbol{0}_{s_A} \end{bmatrix} \Delta \text{t} \right) \begin{bmatrix} \boldsymbol{0}_{s_A} \\ I_{s_A} \end{bmatrix}, \end{aligned} \tag{3.25}$$

where expm$(\cdot)$ computes the matrix exponential. The derivation of (3.25) is in Appendix 3.B. This approach is inspired by the *matrix fraction decomposition* technique, which was utilised for solving *matrix Riccati differential equation* in [122] and in [96] to compute the Gaussian covariance matrix. The latter matrix has the same form as $\boldsymbol{Q}(\Delta t)$ and $\boldsymbol{\Omega}(\Delta t)$ in our model. The scheme in [96] is used to to compute $\boldsymbol{Q}(\Delta t)$,

leading to

$$\begin{aligned}\boldsymbol{Q}(\Delta t) =& \boldsymbol{J}_Q \boldsymbol{K}_Q^{-1} \\ \begin{bmatrix}\boldsymbol{J}_Q \\ \boldsymbol{K}_Q\end{bmatrix} =& \text{expm}\left(\begin{bmatrix}\boldsymbol{A} & \boldsymbol{H}\boldsymbol{H}^\top \\ \boldsymbol{0}_{s_A} & -\boldsymbol{A}^\top\end{bmatrix}\Delta \text{t}\right)\begin{bmatrix}\boldsymbol{0}_{s_A} \\ I_{s_A}\end{bmatrix}\end{aligned} \tag{3.26}$$

see [96] for the proof. Note that the $\boldsymbol{\Omega}(\Delta t)$ in (3.23) only differs from the $\boldsymbol{Q}(\Delta t)$ in (3.12) by the mapping matrix, thus we only need to replace $\boldsymbol{H}$ in (3.26) with $\boldsymbol{G}$ to obtain the $\boldsymbol{\Omega}(\Delta t)$.

However, we stress that in the case that these matrix exponentials cannot be computed, an explicit expression of $\boldsymbol{M}(\Delta t), \boldsymbol{Q}(\Delta t),$ and $\boldsymbol{\Omega}(\Delta t)$ should be sought; this can be potentially simplified by the block-diagonal structure of $\boldsymbol{A}$, $\boldsymbol{H}$ and $\boldsymbol{G}$. For instance, with $\boldsymbol{Q}(\Delta t)$ the integral can be expanded as a block diagonal matrix with m integrals, i.e.

$$\boldsymbol{Q}(\Delta t) = diag(\boldsymbol{q}_1, \boldsymbol{q}_2, ..., \boldsymbol{q}_m), \tag{3.27}$$

where $\boldsymbol{q}_j = \int_0^{\Delta t} e^{\boldsymbol{a}_j(\Delta t-u)}\boldsymbol{h}_j\boldsymbol{h}_j^\top e^{\boldsymbol{a}_j(\Delta t-u)^\top} du$, with $j = 1, 2, ...m$, is the integral on $\mathbb{R}^{s_j \times s_j}$ and thereby, we only require the explicit form of $\boldsymbol{q}_j$; this can be found in the literature for some of the models discussed below, e.g. for Singer model in [93] and Langevin model in [21].

Next, we present four examples of the proposed stable Lévy modelling approach. They are divided into two classes, those for manoeuvring object (sensor-level) tracking in Section 3.4.1 and intent inference in Section 3.4.2. They all have isotropic linear setup such that the parameters $\boldsymbol{a}_j$ and $\boldsymbol{h}_j$ in (3.3) as well as $\boldsymbol{g}_j$ in (3.20) are constants across the different dimensions j, $j = 1, 2, ..., m$. The j-th dimensional target state $\boldsymbol{x}_j$ also incorporates the same kind of kinematics for different dimensions. For notational brevity, we further define x_j as the position in the j-th dimension, $\dot{x}_j$ as its first derivative (i.e. velocity) and $\ddot{x}_j$ is the second derivative of x_j (i.e. acceleration). While we present just four examplar model structures, it should be emphasised that any other linear state space models in continuous time can be adapted to the Lévy state space setting.

3.4.1 Models for manoeuvring object tracking

Below, we consider two models for sensor-level tracking, both are driven by a single noise model as in Section 3.3.1.

Table 3.1 Summary of the parameters in the SDEs (3.2) for Langevin, Singer and equilibrium reverting velocity (ERV) models and (3.19) for the latent destination model, $j = 1, 2, ..., m$.

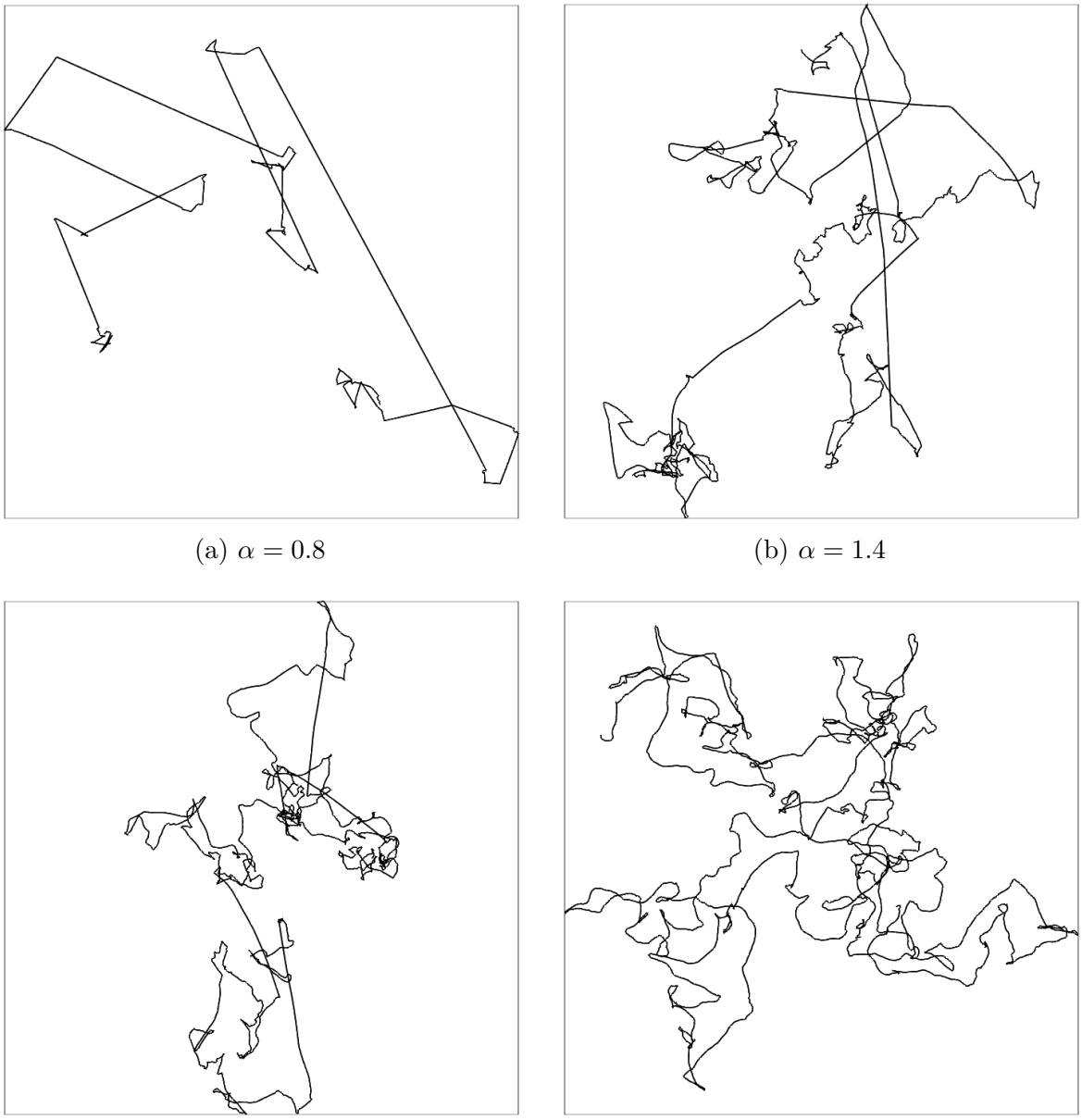
Models	$\boldsymbol{x}_j$	$\boldsymbol{a}_j$	$\boldsymbol{b}_j$	$\boldsymbol{h}_j$	$\boldsymbol{g}_j$
Langevin	$\begin{bmatrix} x_j \\ \dot{x}_j \end{bmatrix}$	$\begin{bmatrix} 0 & 1 \\ 0 & -\lambda \end{bmatrix}$	$\mathbf{0}$	$\begin{bmatrix} 0 \\ 1 \end{bmatrix}$	N/A
Singer	$\begin{bmatrix} x_j \\ \dot{x}_j \\ \ddot{x}_j \end{bmatrix}$	$\begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & -\lambda \end{bmatrix}$	$\mathbf{0}$	$\begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$	N/A
ERV	$\begin{bmatrix} x_j \\ \dot{x}_j \end{bmatrix}$	$\begin{bmatrix} 0 & 1 \\ -\eta & -\rho \end{bmatrix}$	$\eta \begin{bmatrix} 0 \\ p_j \end{bmatrix}$	$\begin{bmatrix} 0 \\ 1 \end{bmatrix}$	N/A
Latent destination	$\begin{bmatrix} x_j \\ \dot{x}_j \\ l_j \end{bmatrix}$	$\begin{bmatrix} 0 & 1 & 0 \\ -\eta & -\rho & \eta \\ 0 & 0 & 0 \end{bmatrix}$	$\mathbf{0}$	$\begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$	$\begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$

3.4.1.1 Stable Lévy Langevin model

This replaces the Brownian motion in the original Langevin dynamics with the isotropic stable Lévy motion. It can be described by the SDE in (3.2), with the parameters listed in Table 3.1. Such parameters correspond to a SDE for the object speed of the j-th spatial dimension according to

$$d\dot{x}_j(t) = -\lambda \dot{x}_j(t)dt + dW_j(t), \tag{3.28}$$

where λ is the positive friction constant and $W_j(t)$ is the j-th component of $\boldsymbol{W}(t)$. This SDE describes $\dot{x}_j(t)$ as the Lévy *Ornstein-Uhlenbeck* process with the mean term 0; hence a target governed by this SDE always tends to slow down despite the fluctuations induced by the Lévy stable noise. It is noted that when the stable noise does not exhibit extreme values, the object displays small non-directional movements since the velocity is near zero. The trajectory is more likely to cluster in a small region during these periods. However, a large velocity can be generated due to an extreme noise value(s) producing a target motion along a straight-line, albeit the velocity decaying gradually, see Fig. 3.1 for example synthetic tracks. Hence, the target occasionally moves fast to other spatial regions, which can capture the efficient foraging/searching pattern of animals. Consequently, this Langevin model is expected to be able to model irregular manoeuvres of highly volatile objects, e.g. animals or insects [97, 116–118, 98].



(a) $\alpha = 0.8$ (b) $\alpha = 1.4$

(c) $\alpha = 1.7$ (d) $\alpha = 2$ (Gaussian case)

Fig. 3.1 Synthetic data generated with the planar Langevin model.

Here we present several synthetic tracks generated by the planar stable Lévy Langevin model to showcase its potential in modelling the irregular manoeuvres of highly volatile objects. Specifically, Fig. 3.1 presents the synthetic data (generated and visualised using MATLAB) from the planar Langevin model with different tail parameters α. The four sub-figures are generated using the same time length and the same parameters with the only exception of α. In particular, we simulate 10000 time steps with $\lambda = 0.0145$, $\sigma_W = 0.033$, $c = 20$, and time interval 1. Here, a small value of

λ is used to ensure slower velocity decay following a heavy-tailed noise disturbance, providing ample data points to capture the resulting near-straight trajectory. Note that the labels on the axis are omitted to avoid any misunderstanding caused by different scale parameters. It can be seen from Fig. 3.1 that the sharp manoeuvres (presented as straight lines) dominate other small movements when $\alpha = 0.8$. However, the manoeuvres are less frequent and smaller as α becomes larger. Ultimately, in the Gaussian case (i.e. $\alpha = 2$), the generated path is relatively rougher and no distinctive manoeuvres can be seen. These examples demonstrate the flexibility of Lévy Langevin model for manoeuvring behaviours modelling through different choices of α.

In a special case of $\lambda = 0$, the parameter setting of the Langevin model becomes that of the constant velocity model, which is widely adopted in the object tracking field. In this case, our stable Lévy formulation introduces a heavy-tailed noise in this canonical (and in a sense non-manoeuvring [10]) dynamical model to characterise the abrupt state change induced by manoeuvres.

3.4.1.2 Stable Lévy Singer model

This model is based on the well-known Gaussian Singer dynamic model in [93] for manoeuvring object tracking. Here we replace the Gaussian noise with an α-stable noise for more flexibility in representing increased object manoeuvrability. It follows the SDE in (3.2) on page 59 and its parameters are listed in Table 3.1. This extends the previous Langevin model to a higher order kinematics and its acceleration follows the Lévy *Ornstein-Uhlenbeck* process with mean term 0. Its j-th dimensional SDE is

$$d\ddot{x}_j(t) = -\lambda \ddot{x}_j(t)dt + dW_j(t). \tag{3.29}$$

The manoeuvres are introduced in this model by the non-zero acceleration and the positive parameter λ characterises the reversion ability from undertaking manoeuvring movements. For instance, a small λ is for a slower decaying of the manoeuvring action, i.e. the manoeuvring lasts longer. Demonstration of this Singer model for vessel tracking is presented in the Section 3.6.3. A special case of the stable Lévy Singer model is when $\lambda = 0$, which results in the stable Lévy constant acceleration model, whose Gaussian noise form is also widely adopted for manoeuvring object tracking.

3.4.2 Models for intent inference

In this section, two models for the meta-level tracking task of destination prediction are presented.

3.4.2.1 Stable Lévy equilibrium reverting velocity model

This spatial model is a modified version of the *equilibrium reverting velocity* (ERV) model developed in [67] and reviewed in Section 2.3.1.1 on page 24 for 3-D pointing gesture movements. The single Gaussian noise of the ERV model in [67] is replaced by a stable Lévy noise to better handle severely perturbed motion. As described in Section 3.3.1, it is a single noise driven model expressed by the SDE in (3.2). Its parameters are given in Table 3.1 such that η and ρ are positive reversion and damping coefficients, respectively. In this version of the model, vector $[p_1, p_2, ..., p_m]^\top$ denotes the destination position. The j-th dimensional SDE is

$$d\dot{x}_j(t) = \eta\left(p_j - x_j(t)\right)dt - \rho\dot{x}_j(t)dt + dW_j(t). \tag{3.30}$$

This SDE describes a target that reverts to the destination's position $[p_1, p_2, ..., p_m]^\top$ under the impact of disturbances of a stable Lévy noise. The damping coefficient ρ ensures that the target speed does not become excessive high during its journey to/at the endpoint. This spatial stable Lévy ERV model is designed to capture the dependence of the destination and the target's movements, and hence it is suitable for destination-aware tracking and/or intent inference. Example synthetic tracks generated by this Lévy ERV model for modelling pointing motion will be provided in Section 3.6.2. It is emphasised that any other Ornstein-Uhlenbeck-based destination reverting models in [69, 67], can be utilised similarly to the ERV, with a stable Lévy dynamic noise in lieu of their original Gaussian noise.

3.4.2.2 Latent destination model

Unlike the aforementioned ERV model which assumes that the intent/destination is a fixed (certain) parameter throughout the target journey, here we model the destination as a latent variable. This hidden endpoint can either be static or dynamic (i.e. follows a Brownian motion), while the tracked target (i.e. its kinematic state) still reverts to the latent endpoint with similar dynamic characteristics as the stable Lévy ERV model. Thereby, this Lévy dynamic/static latent destination model (abbreviated as DLD/SLD model) belongs to the family of models driven by a stable Lévy noise and Gaussian noise as in Section 3.3.2, and follows the SDE in (3.19). Its parameters are defined in Table 3.1, where l_j is the position of the intended destination in the j-th

dimension. The SDE of the motion in the j-th dimension can be written as

$$\begin{aligned} d\dot{x}_j(t) =& \eta \left(l_j(t) - x_j(t)\right) dt - \rho \dot{x}_j(t) dt + dW_j(t), \\ dl_j(t) =& d\beta_j(t), \end{aligned} \tag{3.31}$$

where $\beta_j(t)$ is the j-th component of the isotropic Brownian motion $\boldsymbol{\beta}(t)$. By modelling the intent as a Brownian motion, the DLD model allows a change of destination with a moderate rate. Additionally, by setting the Brownian motion covariance matrix as zero, i.e. $\sigma_B = 0$, this SDE describes the SLD model, and characterises an object reverting to a fixed (*a priori* unknown) destination. However, this SLD model still allows us to estimate the intent with a Gaussian prior on a potential destination position. Most importantly, with this novel modelling approach, we can determine the probability of any point/region within the surveyed area being the target intended endpoint, unlike the formulations in [68–70] that explicitly assume a finite set of nominal destinations. Practically, this model serves as the extension of the ERV or any other similar destination-reverting model to effectively tackle the problem of a dynamically changing intent or the number of possible destinations is too large. This is besides the α-stable dynamic noise aspect. We show in Section 3.5.2 that the intent prediction with this latent destination model is substantially more efficient than the stable Lévy ERV model.

3.5 Estimation and intent inference

Owing to the conditionally Gaussian structure of the stable Lévy model detailed above, standard state and parameters estimation procedures for Gaussian models can be employed within the introduced sequential Monte Carlo framework. Specifically, an efficient Rao-Blackwellised particle filter can be designed to carry out the state estimation, with a simple linear Gaussian observation function. Here, we first discuss computing the likelihoods and posteriors conditioned on the latent variables $\{\Gamma_i, V_i\}^c$, which will then be utilised in the Rao-Blackwellised particle filter for marginal state estimation and intent inference application.

Let $\boldsymbol{y}(t)$ be a vector of observations received at time t. We assume that the sensory observations are available at each of the successive discrete time points t_n $(n = 1, 2, ...)$. Additionally, we define $\boldsymbol{x}_n = \boldsymbol{x}(t_n)$, $\boldsymbol{y}_n = \boldsymbol{y}(t_n)$, and $\{\Gamma_i, V_i\}^c_{n:m}$ as all the latent variables required for state transition from t_n to t_m $(m > n)$, i.e. $\{\Gamma_i, V_i\}^c_{n:m} = \bigcup_{k=n}^{m-1} \{\Gamma_i, V_i\}^c_{\Delta t_k}$, with $\Delta t_k = t_{k+1} - t_k$. Note that for different n, all $\{\Gamma_i, V_i\}^c_{n-1:n}$s are defined in

non-overlapping time intervals, and thus they are independent by construction. The transition density for $\{\Gamma_i, V_i\}^c$ can then be simplified as follows,

$$p(\{\Gamma_i, V_i\}^c_{n-1:n} | \{\Gamma_i, V_i\}^c_{1:n-1}) = p(\{\Gamma_i, V_i\}^c_{n-1:n}). \tag{3.32}$$

We aim to calculate the conditional likelihoods and posteriors, e.g. $p(\boldsymbol{x}_n | \{\Gamma_i, V_i\}^c_{1:n}, \boldsymbol{y}_{1:n})$, in closed forms. In a typical setup where the noise parameter σ_W^2 is a certain constant, this requires the following observation function,

$$p(\boldsymbol{y}_n | \boldsymbol{x}_n) = \mathcal{N}(\boldsymbol{y}_n; \boldsymbol{C}\boldsymbol{x}_n + \boldsymbol{D}, \boldsymbol{R}), \tag{3.33}$$

where $\boldsymbol{R}$ is the covariance of the observation noise; $\boldsymbol{C}$ and $\boldsymbol{D}$ are parameters defining the transformation from state to the observations. Consequently, the sought conditional likelihoods and posteriors can be recursively computed by the standard Kalman filter with Gaussian priors, i.e.

$$p(\boldsymbol{x}_n | \{\Gamma_i, V_i\}^c_{1:n}, \boldsymbol{y}_{1:n}) = \mathcal{N}(\boldsymbol{x}_n; \boldsymbol{\mu}_{n|n}, \boldsymbol{P}_{n|n}), \tag{3.34}$$

$$p(\boldsymbol{x}_n | \{\Gamma_i, V_i\}^c_{1:n}, \boldsymbol{y}_{1:n-1}) = \mathcal{N}(\boldsymbol{x}_n; \boldsymbol{\mu}_{n|n-1}, \boldsymbol{P}_{n|n-1}), \tag{3.35}$$

$$p(\boldsymbol{y}_n | \{\Gamma_i, V_i\}^c_{1:n}, \boldsymbol{y}_{1:n-1}) = \mathcal{N}(\boldsymbol{y}_n; \hat{\boldsymbol{y}}_{n|n-1}, \boldsymbol{\Sigma}_{n|n-1}), \tag{3.36}$$

with means and covariances given by

$$\begin{aligned}
\Delta t &= t_n - t_{n-1}, \\
\boldsymbol{\mu}_{n|n-1} &= \boldsymbol{F}(\Delta t)\boldsymbol{\mu}_{n-1|n-1} + \boldsymbol{M}(\Delta t), \\
\boldsymbol{P}_{n|n-1} &= \boldsymbol{F}(\Delta t)\boldsymbol{P}_{n-1|n-1}\boldsymbol{F}(\Delta t)^\top + \boldsymbol{S}(\Delta t), \\
\hat{\boldsymbol{y}}_{n|n-1} &= \boldsymbol{C}\boldsymbol{\mu}_{n|n-1} + \boldsymbol{D}, \\
\boldsymbol{\Sigma}_{n|n-1} &= \boldsymbol{C}\boldsymbol{P}_{n|n-1}\boldsymbol{C}^\top + \boldsymbol{R}, \\
\boldsymbol{K} &= \boldsymbol{P}_{n|n-1}\boldsymbol{C}^\top\boldsymbol{\Sigma}^{-1}_{n|n-1}, \\
\boldsymbol{\mu}_{n|n} &= \boldsymbol{\mu}_{n|n-1} + \boldsymbol{K}(\boldsymbol{y}_n - \hat{\boldsymbol{y}}_{n|n-1}), \\
\boldsymbol{P}_{n|n} &= (I_{s_A} - \boldsymbol{K}\boldsymbol{C})\boldsymbol{P}_{n|n-1},
\end{aligned} \tag{3.37}$$

where $\boldsymbol{S}(\Delta t)$ is defined in (3.18) on page 64 for single noise driven model and in (3.22) on page 65 for non-Gaussian α-stable and Gaussian noises driven case. It should be noted that without the linear Gaussian observation function in (3.33), the closed form results in (3.34)-(3.36) will be lost and one may have to sample both the state $\boldsymbol{x}$ as well as the latent variables $\{\Gamma_i, V_i\}$ in the particle filtering for state estimation.

Furthermore, it is possible to jointly and explicitly estimate the noise parameter σ_W^2 and the state conditioned on the latent variables, i.e. $p(\sigma_W^2, \boldsymbol{x}_n|\{\Gamma_i, V_i\}_{1:n}^c, \boldsymbol{y}_{1:n})$. Such a tractable estimation is only valid for the single noise driven stable Lévy models in Section 3.3.1, and imposes two more assumptions, i.e. $\boldsymbol{R}$ is scaled with σ_W^2 and the prior $p(\sigma_W^2)$ is the inverted gamma distribution. Subsequently, the standard Kalman filter can still be employed to evaluate the joint posterior in a closed form. Relevant details of this can be found in [21]. In this chapter, we do not treat such a scenario and assumes that σ_W^2 is known.

3.5.1 Rao-Blackwellised particle filtering

Based on the above closed-form conditional densities, the efficient Rao-Blackwellised particle filter can be used to only sample the intractable latent variables $\{\Gamma_i, V_i\}^c$ and other marginal densities, e.g. $p(\boldsymbol{x}_n|\boldsymbol{y}_{1:n})$, can be estimated within the importance sampling framework. Therefore, we consider the posterior of $\{\Gamma_i, V_i\}^c$ as the particle filter target distribution, which can be factorised as per,

$$\begin{aligned} p(\{\Gamma_i, V_i\}_{1:n}^c|\boldsymbol{y}_{1:n}) \propto & p(\{\Gamma_i, V_i\}_{n-1:n}^c|\{\Gamma_i, V_i\}_{1:n-1}^c) \\ & \times p(\{\Gamma_i, V_i\}_{1:n-1}^c|\boldsymbol{y}_{1:n-1}) p(\boldsymbol{y}_n|\{\Gamma_i, V_i\}_{1:n}^c, \boldsymbol{y}_{1:n-1}), \end{aligned} \tag{3.38}$$

such that the transition for $\{\Gamma_i, V_i\}^c$ can be simplified as in (3.32). With $N_{\mathcal{P}}$ particles, then at time t_n each particle is a set of latent variables sequence $\{\Gamma_i, V_i\}_{1:n}^{c,(p)}$, where p is the index of particles. We now define the *unnormalised* importance weight for the p-th particle as $\omega_n^{(p)}$, i.e.

$$\omega_n^{(p)} = \frac{p(\{\Gamma_i, V_i\}_{1:n}^{c,(p)}|\boldsymbol{y}_{1:n})}{q(\{\Gamma_i, V_i\}_{1:n}^{c,(p)}|\boldsymbol{y}_{1:n})},$$

and q is the proposal distribution. Let the *self-normalised* importance weight be $\omega_n^{*(p)} = \omega_n^{(p)} / \sum_{p=1}^{N_{\mathcal{P}}} \omega_n^{(p)}$. Such *self-normalised* weights offer an empirical Dirac approximation of the target distribution, i.e.

$$p(\{\Gamma_i, V_i\}_{1:n}^c|\boldsymbol{y}_{1:n}) \approx \sum_{p=1}^{N_{\mathcal{P}}} \omega_n^{*(p)} \delta(\{\Gamma_i, V_i\}_{1:n}^{c,(p)}). \tag{3.39}$$

To update the *self-normalised* importance weight, we expand it as follows,

$$\begin{aligned}\omega_n^{*(p)} \propto \omega_n^{(p)} &= \frac{p(\{\Gamma_i, V_i\}_{1:n}^{c,(p)}|\boldsymbol{y}_{1:n})}{q(\{\Gamma_i, V_i\}_{1:n}^{c,(p)}|\boldsymbol{y}_{1:n})} \\ &\propto \frac{p(\{\Gamma_i, V_i\}_{1:n-1}^{c,(p)}|\boldsymbol{y}_{1:n-1})}{q(\{\Gamma_i, V_i\}_{1:n-1}^{c,(p)}|\boldsymbol{y}_{1:n-1})} \frac{p(\{\Gamma_i, V_i\}_{n-1:n}^{c,(p)}|\{\Gamma_i, V_i\}_{1:n-1}^{c,(p)})}{q(\{\Gamma_i, V_i\}_{n-1:n}^{c,(p)}|\{\Gamma_i, V_i\}_{1:n-1}^{c,(p)}, \boldsymbol{y}_n)} p(\boldsymbol{y}_n|\{\Gamma_i, V_i\}_{1:n}^{c,(p)}, \boldsymbol{y}_{1:n-1}) \\ &\propto \omega_{n-1}^{*(p)} p(\boldsymbol{y}_n|\{\Gamma_i, V_i\}_{1:n}^{c,(p)}, \boldsymbol{y}_{1:n-1}),\end{aligned} \tag{3.40}$$

where we choose the target transition density in (3.32) as the proposal, i.e. the *bootstrap* particle filter. The sampling scheme for such a proposal is summarised in Algorithm 3. The conditional predictive likelihood in (3.40) can be computed directly using the Kalman filter steps in (3.36).

Algorithm 3: Sampling scheme for latent variables

Output: $\{\Gamma_i, V_i\}_{n-1:n}^{c}$.
$\Delta t = t_n - t_{n-1}$.
$i = 1$, sample $\Gamma_i \sim \exp(1)$.
while $\Gamma_i < c\Delta t$ **do**
    Sample $V_i \sim$ *Uniform*$(0, \Delta t)$.
    Include Γ_i and V_i to $\{\Gamma_i, V_i\}_{n-1:n}^{c}$.
    $i \leftarrow i + 1$; sample Γ_i such that $(\Gamma_i - \Gamma_{i-1}) \sim \exp(1)$.
end

Consequently, we can recursively estimate the state and the nonlinear latent variables $\{\Gamma_i, V_i\}^c$ in an efficient Rao-Blackwillisation scheme, with each particle updating and storing the following information: the sample $\{\Gamma_i, V_i\}^c$, normalised weight ω^*, the conditional state mean $\boldsymbol{\mu}$ and the covariance $\boldsymbol{P}$. This filtering strategy is outlined in Algorithm 4. With this stored information, the estimation for other marginal densities can be extracted as required. Specifically,

$$\begin{aligned}p(\cdot|\boldsymbol{y}_{1:n}) &= \int p(\cdot|\boldsymbol{y}_{1:n}, \{\Gamma_i, V_i\}_{1:n}^{c}) p(\{\Gamma_i, V_i\}_{1:n}^{c}|\boldsymbol{y}_{1:n}) d\{\Gamma_i, V_i\}_{1:n}^{c} \\ &\approx \sum_{p=1}^{N_\mathcal{P}} \omega_n^{*(p)} p(\cdot|\boldsymbol{y}_{1:n}, \{\Gamma_i, V_i\}_{1:n}^{c,(p)}),\end{aligned} \tag{3.41}$$

where we substitute (3.39) to obtain the third line. Note that '$\cdot$' can be any of the variables which have explicit conditional densities, such as $\boldsymbol{x}_n$ as given in (3.34)-(3.36). For example, the marginal posterior of the target state, which may be required in

object tracking, can be approximated by (3.41) as,

$$p(\boldsymbol{x}_n|\boldsymbol{y}_{1:n}) \approx \sum_{p=1}^{N_\mathcal{P}} \omega_n^{*(p)} p(\boldsymbol{x}_n|\boldsymbol{y}_{1:n}, \{\Gamma_i, V_i\}_{1:n}^{c,(p)}), \tag{3.42}$$

such that the conditional posterior $p(\boldsymbol{x}_n|\boldsymbol{y}_{1:n}, \{\Gamma_i, V_i\}_{1:n}^{c,(p)})$ is given in (3.34).

Algorithm 4: Single iteration of particle filtering

Input: $\{\Gamma_i, V_i\}_{1:n-1}^{c,(p)}, \omega_{n-1}^{*(p)}$, and $\{\boldsymbol{\mu}, \boldsymbol{P}\}_{n-1|n-1}^{(p)}$.
Output: $\{\Gamma_i, V_i\}_{1:n}^{c,(p)}, \omega_n^{*(p)}$ and $\{\boldsymbol{\mu}, \boldsymbol{P}\}_{n|n}^{(p)}$.
if *Resample* **then**
  Resample particles and set *self-normalised* weights $\omega_{n-1}^{*(p)} = 1/N_\mathcal{P}$.
end
for *particles* $p = 1 : N_\mathcal{P}$ **do**
  Sample $\{\Gamma_i, V_i\}_{n-1:n}^{c,(p)}$ via Algorithm 3, and include it to $\{\Gamma_i, V_i\}_{1:n}^{c,(p)}$.
  Update stored information $\{\boldsymbol{\mu}, \boldsymbol{P}\}_{n|n}^{(p)}$ via (3.37).
  Compute the *self-normalised* weight $\omega_n^{*(p)}$ via (3.40).
end

3.5.2 Intent inference

Similar to the previous chapter, the objective here is to sequentially infer the target intent with recursive Bayesian filtering. For each of the two stable Lévy models in Section 3.4.2, an inference routine is introduced below with its pseudo-code.

3.5.2.1 Intent inference for stable Lévy ERV model

Since the stable Lévy ERV model is an extension of the Gaussian mean reverting model introduced in Chapter 2, the intent in this example is modeled as a finite discrete grid of possible destinations as in Chapter 2. The scenario with infinite number of destinations will be considered later in Section 3.5.2.2. As a result, the Bayesian intent inference framework introduced in Section 2.2.2 from the previous chapter can be directly applied. Specifically, the particle filter is used to estimate the probability of each destination from the sequentially evolving data. In this section, we briefly describe the intent inference strategy, with the details and rationale provided in the Section 2.2.2 from the previous chapter.

Let there be a total of N_D nominal endpoints, indexed by $d = 1, 2, ..., N_D$. The position of each endpoint is known and defined by the vector $\boldsymbol{p}_d = [p_1^d, p_2^d, ..., p_m^d]^\top$.

Our objective is to determine the probability of each endpoint being the intended destination $\mathcal{D} \in \{1, 2, ..., N_D\}$. Here the intended destination $\mathcal{D}$ is treated as an underlying parameter of the dynamic model. Subsequently, the goal of destination prediction can be rephrased as computing the probability of $\mathcal{D} = d$ for each nominal endpoint $d = 1, 2, ..., N_D$. The Bayesian formula allows us to factorise this computation as follows:

$$p(\mathcal{D} = d|\boldsymbol{y}_{1:n}) \propto p(\boldsymbol{y}_{1:n}|\mathcal{D} = d)p(\mathcal{D} = d). \tag{3.43}$$

The prior $p(\mathcal{D} = d)$ is chosen according to the contextual information (e.g. travel history, preferences, etc.). It can be uniform, i.e. $p(\mathcal{D} = d) = 1/N_{\mathcal{D}}$, for all nominal endpoints $d = 1, 2, ..., N_D$ if there is no such information is available. The index of the most probable destination, in the *Maximum a Posteriori* sense, is

$$i = \underset{d=1,2,...,N_{\mathcal{D}}}{\arg\max}\ p(\mathcal{D} = d|\boldsymbol{y}_{1:n}). \tag{3.44}$$

The destination likelihood in (3.43) can be recursively evaluated via the *prediction error decomposition* (PED) [87],

$$p(\boldsymbol{y}_{1:n}|\mathcal{D} = d) = p(\boldsymbol{y}_n|\boldsymbol{y}_{1:n-1}, \mathcal{D} = d)p(\boldsymbol{y}_{1:n-1}|\mathcal{D} = d). \tag{3.45}$$

Thus, the key operation is computing the predictive likelihood $p(\boldsymbol{y}_n|\boldsymbol{y}_{1:n-1}, \mathcal{D} = d)$. Here we define the destination driven motion model (conditioned on $\mathcal{D} = d$) to be the stable Lévy ERV model. This is feasible since the model is designed to revert to the d-th endpoint position $\boldsymbol{p}_d$ with the parameters specified in Table 3.1. Thereby, the sought predictive likelihood can be approximated in a manner analogous to (2.42) from Chapter 2, using the importance sampling framework [43, 46]

$$\begin{aligned} p(\boldsymbol{y}_n|\boldsymbol{y}_{1:n-1}, \mathcal{D} = d) &\approx \sum_{p=1}^{N_{\mathcal{P}}} \omega_{n-1}^{*(p,d)} p(\boldsymbol{y}_n|\{\Gamma_i, V_i\}_{1:n}^{c,(p,d)}, \boldsymbol{y}_{1:n-1}, \mathcal{D} = d) \\ &= \sum_{p=1}^{N_{\mathcal{P}}} \tilde{\omega}_n^{(p,d)}, \end{aligned} \tag{3.46}$$

where the superscript d stipulates that the variable/sample belongs to the d-th destination conditioned ERV model. In the last step of (3.46) the *update* weight is defined by

$$\tilde{\omega}_n^{(p,d)} = \omega_{n-1}^{*(p,d)} p(\boldsymbol{y}_n|\{\Gamma_i, V_i\}_{1:n}^{c,(p,d)}, \boldsymbol{y}_{1:n-1}, \mathcal{D} = d). \tag{3.47}$$

Since, by definition, the destination information is an implicit condition for a stable Lévy mean reverting model, the conditional predictive likelihood in (3.47) is in effect (3.36). Note that $\tilde{\omega}_n^{(p,d)}$ is also required in the *self-normalised* weight update equation (3.40). Subsequently, we have

$$\omega_n^{*(p,d)} \propto \tilde{\omega}_n^{(p,d)}. \tag{3.48}$$

For $N_{\mathcal{D}}$ possible destinations, a total of $N_{\mathcal{D}}$ Rao-Blackwellised particle filter filters, each with a particular destination information, would be required to carry out the intent inference. By only sampling $\{\Gamma_i, V_i\}^c$ once, here all $N_{\mathcal{D}}$ particle filters share the same Lévy noise latent samples. This not only reduces the time for sampling latent variables, but also for computation of quantities like $\boldsymbol{S}(\Delta t)$ required in (3.37). Whereas this sampling and computation were formerly executed for every individual destination, it is now performed just once, which significantly reduces the computation time for a large number of destinations. The inference procedure for the stable Lévy mean reverting model is outlined in Algorithm 5. This pseudo-code only proceeds one time step from t_{n-1} to t_n up to the current observation y_n. More details about this intent inference strategy can be found in [69] and the previous chapter, where a similar conditionally Gaussian intent inference framework is presented for jump models.

3.5.2.2 Intent inference for latent destination models

Since the position of the intended destination is modelled as a variable within the dynamical state (see Table 3.1), the estimated intent can be directly extracted from the estimated "extended" dynamical state employing (3.41), i.e.

$$\begin{aligned} p(\boldsymbol{l}_n|\boldsymbol{y}_{1:n}) =& p(\boldsymbol{G}^\top \boldsymbol{x}_n|\boldsymbol{y}_{1:n}) \\ \approx& \sum_{p=1}^{N_{\mathcal{P}}} \omega_n^{*(p)} p(\boldsymbol{G}^\top \boldsymbol{x}_n|\boldsymbol{y}_{1:n}, \{\Gamma_i, V_i\}_{1:n}^{c,(p)}) \\ =& \sum_{p=1}^{N_{\mathcal{P}}} \omega_n^{*(p)} \mathcal{N}(\boldsymbol{l}_n; \boldsymbol{G}^\top \boldsymbol{\mu}_{n|n}^{(p)}, \boldsymbol{G}^\top \boldsymbol{P}_{n|n}^{(p)} \boldsymbol{G}), \end{aligned} \tag{3.49}$$

where the last line is obtained by substituting the conditional state posterior in (3.34) and using change of variables for Gaussian vectors. Vector $\boldsymbol{l}_n$ denotes the position of the intent/endpoint at time t_n, i.e. $\boldsymbol{l}_n = [l_1(t_n), l_2(t_n), ..., l_m(t_n)]^\top$. The mapping matrix $\boldsymbol{G}$ is given for the latent destination models in (3.20) as per Table 3.1. The mean $\boldsymbol{\mu}_{n|n}^{(p)}$ and covariance $\boldsymbol{P}_{n|n}^{(p)}$, both defined in (3.34), are stored in the particle filter as described in Algorithm 4. One can see that the estimated intent position is a mixture

of Gaussian densities, which has support $\boldsymbol{l}_n \in \mathbb{R}^m$ covering the entire area of interest (e.g. a surveyed region).

The support of the intended destination position can be more restrictive in practice. For example, consider the intent inference task described above for stable Lévy ERV model and in the previous chapter, where the target's destination can only be one of the $N_{\mathcal{D}}$ nominal endpoints. Based on the estimated intent in (3.49), there are then several ways to decide which nominal endpoint is the intended one. Here, we present two methods to determine the most probable destination, whose index is denoted by i. Specifically, the most probable destination is the one that: a) achieves the smallest Euclidean distance to the mean of the estimated intent, i.e.

$$i = \underset{d=1,2,\ldots,N_{\mathcal{D}}}{\arg\min} \; \|\mathrm{E}(\boldsymbol{l}_n|\boldsymbol{y}_{1:n}) - \boldsymbol{p}_d\|, \tag{3.50}$$

Algorithm 5: Single iteration (t_{n-1} to t_n) of intent inference with the stable Lévy ERV model

Input: $\omega_{n-1}^{*(p,d)}$, $p(\boldsymbol{y}_{1:n-1}|\mathcal{D}=d)$, and $\{\boldsymbol{\mu}, \boldsymbol{P}\}_{n-1|n-1}^{(p,d)}$.
Output: $\omega_{n}^{*(p,d)}$, $p(\boldsymbol{y}_{1:n}|\mathcal{D}=d)$, and $\{\boldsymbol{\mu}, \boldsymbol{P}\}_{n|n}^{(p,d)}$.
for *particles* $p = 1 : N_{\mathcal{P}}$ **do**
    Sample $\{\Gamma_i, V_i\}_{n-1:n}^{c,(p)}$ via Algorithm 3.
end
for *endpoints* $d = 1 : N_{\mathcal{D}}$ **do**
    if *Resample* **then**
        Resample particles and set *self-normalised* weights $\omega_{n-1}^{*(p,d)} = 1/N_{\mathcal{P}}$.
    end
    for *particles* $p = 1 : N_{\mathcal{P}}$ **do**
        Set $\{\Gamma_i, V_i\}_{n-1:n}^{c,(p,d)} = \{\Gamma_i, V_i\}_{n-1:n}^{c,(p)}$.
        Update stored information $\{\boldsymbol{\mu}, \boldsymbol{P}\}_{n|n}^{(p,d)}$ via (3.37).
        Compute the *update* weight $\tilde{\omega}_n^{(p,d)}$ via (3.47).
    end
    Compute $p(\boldsymbol{y}_n|\boldsymbol{y}_{1:n-1}, \mathcal{D}=d)$ according to (3.46).
    Normalise particles weights $\omega_n^{*(p,d)}$ via (3.48).
    Evaluate endpoint likelihood $p(\boldsymbol{y}_{1:n}|\mathcal{D}=d)$ in (3.45).
end
Determine endpoint probability $p(\mathcal{D}=d|\boldsymbol{y}_{1:n})$ in (3.43).

where the mean can be computed as follows based on (3.49),

$$\mathrm{E}(\boldsymbol{l}_n|\boldsymbol{y}_{1:n}) \approx \sum_{p=1}^{N_{\mathcal{P}}} \omega_n^{*(p)} \boldsymbol{G}^{\top} \boldsymbol{\mu}_{n|n}^{(p)}.$$

or b) attains the highest value in the posterior density of the intent, i.e.

$$i = \underset{d=1,2,\ldots,N_{\mathcal{D}}}{\arg\max}\ p(\boldsymbol{l}_n = \boldsymbol{p}_d|\boldsymbol{y}_{1:n}), \tag{3.51}$$

where the posterior can be directly evaluated according to (3.49). This intent inference strategy for latent destination models is summarised in Algorithm 6. Note that such an algorithm is a standard particle filter procedure with one additional destination-determining step at each time instant. Since these particle filtering steps are additionally necessary for the conventional tracking application, we can regard the estimated intent as a by-product of the (sensor-level) tracking algorithm.

Such a latent destination model intent inference Algorithm 6 has a complexity of $\mathcal{O}(N_{\mathcal{P}}N)$, where N is the number of observations. Consequently, the complexity for ERV intent inference algorithm (i.e. a full iteration of Algorithm 5) should be less than $\mathcal{O}(N_{\mathcal{D}}N_{\mathcal{P}}N)$. This reduction follows from: i) the size of state vector is smaller compared with the latent destination model and ii) the $N_{\mathcal{D}}$ particle filters in the algorithm 5 share the same sample set $\{\Gamma_i, V_i\}^{c,(p)}$. Nonetheless, it should be noted that the computation complexity of the embedded Kalman filter calculations in Algorithm 5 are still relatively high for large number of nominal destinations. Therefore, the latent destination models, which only requires one particle filter, regardless of number of destinations, should be a more computationally efficient approach.

Algorithm 6: Intent inference with the latent destination models

for *particles* $p = 1 : N_{\mathcal{P}}$ **do**
  Set $\{\boldsymbol{\mu}_{0|0}^{(p)}, \boldsymbol{P}_{0|0}^{(p)}\}$ to the mean and covariance of initial state prior $p(\boldsymbol{x}_0)$.
  Set the initial *self-normalised* weight $\omega_0^{*(p)} = \frac{1}{N_{\mathcal{P}}}$.
end
for *time step* $n = 1 : N$ **do**
  Proceed one-step particle filtering via Algorithm 4, and store the output.
  Determine the most probable destination via (3.50) or (3.51).
end

3.6 Results

The performance of the introduced stable Lévy modelling is assessed here with real data from three scenarios. The first two consider the intent inference task for improving human-computer interaction (HCI), specifically when the user point-select motion in 2-D with a mouse cursor and 3-D free hand pointing gestures are subjected to severe perturbations. Knowing the interface item the user intends to select on the interface, early in the point-select task, can noticeably simplify and expedite the interactions, e.g. see the Section 2.1.1 and [123, 90, 7]. In the third case, we use trajectory data from a fast manoeuvring maritime vessel to evaluate the tracking capability (i.e. only kinematic state estimation) of the proposed stable Lévy approach and compare it against other conventional Gaussian models. We also address practical implementation aspects of an stable Lévy model, including parameterisation and complexity.

Unless otherwise specified in subsequent subsections, parameter choices were manually tuned to achieve satisfactory performance across a range of data examples. This follows a similar mixed procedure of initial broad adjustments followed by finer refinements, as detailed in Section 2.5.0.2. It's worth noting that the physical interpretations highlighted in Section 2.5.0.2 were extended to both Gaussian and Lévy ERV models examined in this section. However, one must also recognise the limitations of such physical interpretation-driven tuning, as discussed in the same section. Parameter estimation for these models is an important task that can be carried out using the likelihood functions and/or cross validation [124]; a full exploration of this is beyond the scope of this thesis, which is primarily concerned with the exposition of new methodologies.

3.6.1 Intent inference for HCI with 2-D pointing motion

The employed dataset includes 16 2-D mouse cursor trajectories (each consists of 100-500 data points) recorded by a computer whilst participants suffering from motor-impairment, namely a mild-severe form of cerebral palsy, undertook interaction tasks with a mouse to select icons on experimental graphical user interface (GUI) displayed on the screen; this is the classical Fitt's law task, ISO 9241. The pointing trajectories in this case exhibit tremor and sharp jerks/jolts (i.e. sharp manoeuvres) induced by the motor-impairment. Four example tracks are depicted in Fig. 3.2 with the GUI which comprises of 16 selectable icons in a circular formation. The participant always starts from the central icon prior to selecting a highlighted interface item.

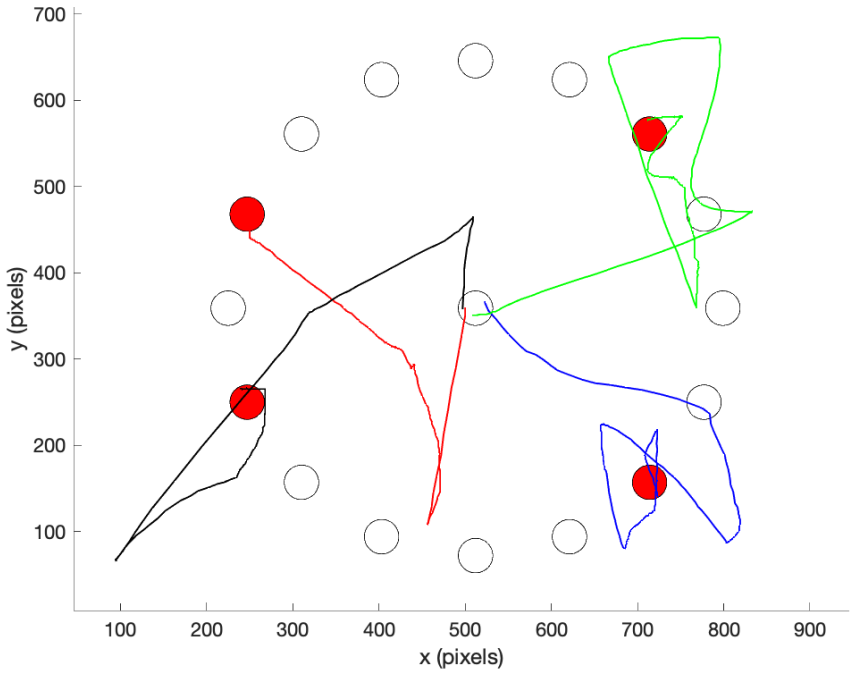


Fig. 3.2 Example real mouse-cursor complete pointing tracks. Four distinct tracks (each plotted in a unique color) originate from the central icon, moving to select a highlighted interface item.

Here, we compare the intent prediction results of several Gaussian and stable-Lévy models from the above pointing dataset. They are the : 1) Gaussian ERV (G-ERV, proposed in [67]), 2) stable Lévy ERV (L-ERV, proposed here in Section 3.4.2.1), 3) Gaussian dynamic ($\sigma_B \neq 0$) latent destination (G-DLD, proposed here similarly as in Section 3.4.2.2), 4) Gaussian static ($\sigma_B = 0$) latent destination (G-SLD, proposed here similarly as in Section 3.4.2.2), 5) stable Lévy DLD (L-DLD, proposed here in Section 3.4.2.2), and 6) stable Lévy SLD (L-SLD, proposed here in Section 3.4.2.2) models.

Performance is evaluated in terms of the aggregate intent inference success rate, defined as the fraction of the pointing motion duration (from the start of the task to selecting the highlighted interface item) for which the true destination is correctly predicted. This metric remains consistent with the one used in the previous chapter. For a more comprehensive understanding of this metric, including its limitations and alternatives, please refer to Section 2.5.0.1. The endpoint is determined according to the decision criteria (3.44) for ERV models and (3.51) for latent destination models. Whilst the standard Kalman filter is utilised for the Gaussian ERV model as in [67] and similar for the G-DLD/G-SLD, Rao-Blackwellised particle filter with 5000 particles is applied with the stable Lévy models and 10 Monte Carlo runs per trajectory are

carried out. The overall average success rates (i.e. average from all 16 tracks and 10 Monte Carlo runs per trajectory for the stable Lévy models) and corresponding standard deviations are shown in Fig. 3.3.

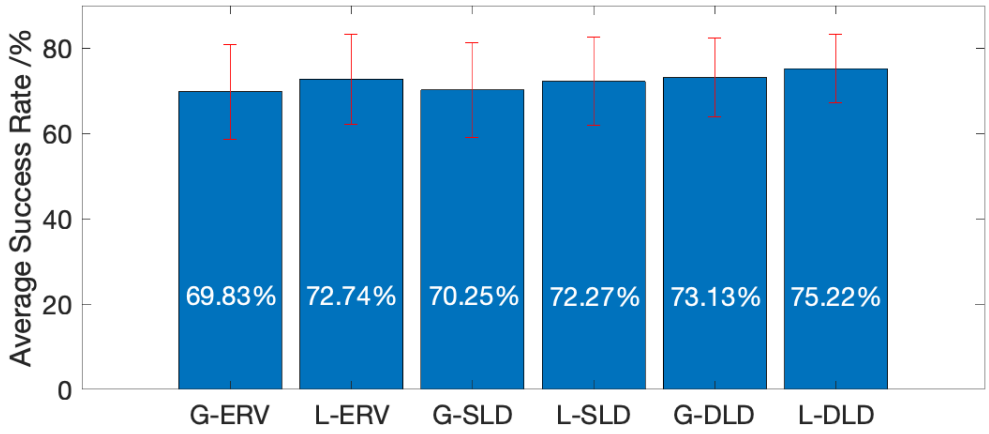


Fig. 3.3 Average intent inference success rate for 2-D pointing tasks; the red error bar indicates ± 1 standard deviation.

The parameters of the considered models are listed in Table 3.2. They were manually tuned to achieve a good individual model performance, which entailed setting different measurement noise covariances, i.e. $\boldsymbol{R}$, albeit for the same (accurate) observations of the cursor location. We also specify the Gaussian prior of intent for the latent destination models, such that its mean is the first observed cursor's position (i.e. in the vicinity of the central GUI icon) and covariance (denoted by $\boldsymbol{\Sigma}_d$) is stated in Table 3.2. For the SLD model, it is necessary that $\boldsymbol{\Sigma}_d$ is sufficiently large to assign reasonable prior probabilities to all nominal endpoints.

It can be noticed from Fig. 3.3 that the proposed stable Lévy models deliver the highest intent prediction success, with the latent destination formulation achieving the best performance. This can be attributed to their ability to better capture the sharp swings and piece-wise straight lines present in the pointing motion (see Fig. 3.2). For the same type of dynamic noise, the success rates of the SLD and ERV models (i.e. G-ERV versus G-SLD or L-ERV versus L-SLD) are similar. This may be due to their common assumption of a fixed intent throughout the pointing task. However, we recall that the latent destination models require lower computational effort with the introduced efficient intent inference procedure. Additionally, the superior results with the DLD models can be due to their intrinsic capacity to quickly recover from a misjudgement on the true endpoint induced by a non-destination reverting (e.g. unintentional perturbation-originated) jump and/or capture the changing nature

Table 3.2 Model parameters

Models	Parameters
G-ERV	$\eta = 0.4$, $\rho = 7$, $\sigma_{Gaus} = 50$, $\boldsymbol{R} = \mathbf{0}$.
L-ERV	$\alpha = 1.45$, $c = 1$, $\eta = 0.4$, $\rho = 1.2$, $\sigma_W = 20$, $\boldsymbol{R} = 25I_2$.
G-SLD	$\eta = 0.4$, $\rho = 8$, $\sigma_{Gaus} = 100$, $\boldsymbol{R} = \mathbf{0}$, $\boldsymbol{\Sigma}_d = 10000I_2$.
L-SLD	$\alpha = 1.6$, $c = 1$, $\eta = 0.3$, $\rho = 1.5$, $\sigma_W = 25$, $\boldsymbol{R} = 25I_2$, $\boldsymbol{\Sigma}_d = 10000I_2$.
G-DLD	$\eta = 0.3$, $\rho = 1.5$, $\sigma_{Gaus} = 35$, $\sigma_B = 10$, $\boldsymbol{R} = \mathbf{0}$, $\boldsymbol{\Sigma}_d = 25I_2$.
L-DLD	$\alpha = 1.5$, $c = 1$, $\eta = 0.3$, $\rho = 1.5$, $\sigma_W = 20$, $\sigma_B = 10$, $\boldsymbol{R} = 25I_2$, $\boldsymbol{\Sigma}_d = 25I_2$.

of intent during a pointing task. For instance, the intent at the beginning of the point-select motion may not be the precise location of the GUI, but rather its general on-screen region or even only its direction. This is supported by the common premise in HCI that a cursor pointing motion consists of two parts, a large ballistic movement to near the endpoint followed by several smaller homing ones [123].

The cursor trajectories in Fig. 3.2 exhibit abrupt jumps/jolts, clearly not following a Gaussian process, and hence more likely to be appropriate for our proposed Lévy models, which can represent more erratic (extreme) manoeuvres. Our models can deal with even more extreme patterns, for instance those induced by a more severe motor impairment condition, as was previously demonstrated in the 2-D synthetic tracks presented in Fig. 3.1.

We would like to emphasise that the performance of all methods shown in Fig. 3.3 greatly exceeds that of a mere random guess, which would have a success rate of about 6.25%. Additionally, if one were to simply predict the nearest icon to the cursor as the intended destination, observation of the four example trajectories in Fig. 3.2 suggests that predictions would be incorrect for over half the track's duration. This is especially noticeable in instances where a sudden movement or jolt redirects the cursor to a distant location, as depicted by the red tracks in Fig. 3.2. In such scenarios, the impressive 75% average success rate of the L-DLD model stands out significantly.

3.6.2 Destination prediction with freehand pointing gestures

Predictive touch technology introduced in Section 2.1.1 utilises observations from a gesture tracker, i.e. the coordinates of the pointing hand/finger in 3-D, to predict (in real-time) the intended GUI icon on a display [67, 7]. In this section, we consider 10 complete 3-D pointing trajectories collected in a moving vehicle by a predictive touch system gesture tracker whilst participants interacted with the main in-car touchscreen

(the Dataset B in Section 2.5). They feature severe perturbations in the form of abrupt manoeuvres and jolts caused by the present vibrations and accelerations due to aggressive driving on badly maintained roads. Fig. 3.4 depicts three of those tracks and the GUI with 37 selectable icons; a track recorded in a static car is included (in red) for reference. Fig. 3.5 also exhibits six "synthetic" trajectories generated by the developed spatial Lévy ERV model to illustrate its potential for representing perturbed intent-driven pointing movements in 3-D. They closely resemble the real data in Fig. 3.4.

Next, we evaluate the intent inference performance of the stable Lévy ERV model and compare it with the outcome of the jump and Gaussian ERV models in [69], which employed the same dataset of 10 real perturbed tracks. The parameters were manually tuned for successful endpoint predictions. For the stable Lévy ERV model they are $\alpha = 1.4$, $\sigma_W = 300$, $c = 20$, $\eta = 60$ and $\rho = 15$; and for the Gaussian ERV model are $\eta = 55$, $\rho = 15$ and $\sigma = 3000$; the jump-ERV model utilises the parameters stated in Table 2.2 and in [69]. Similar to the cursor pointing data in Section 3.6.1, a Gaussian measurement model is assumed.

The prediction success rates for the three assessed models are shown in Fig. 3.6. It can be seen that the proposed Lévy state-space model achieves a significant performance

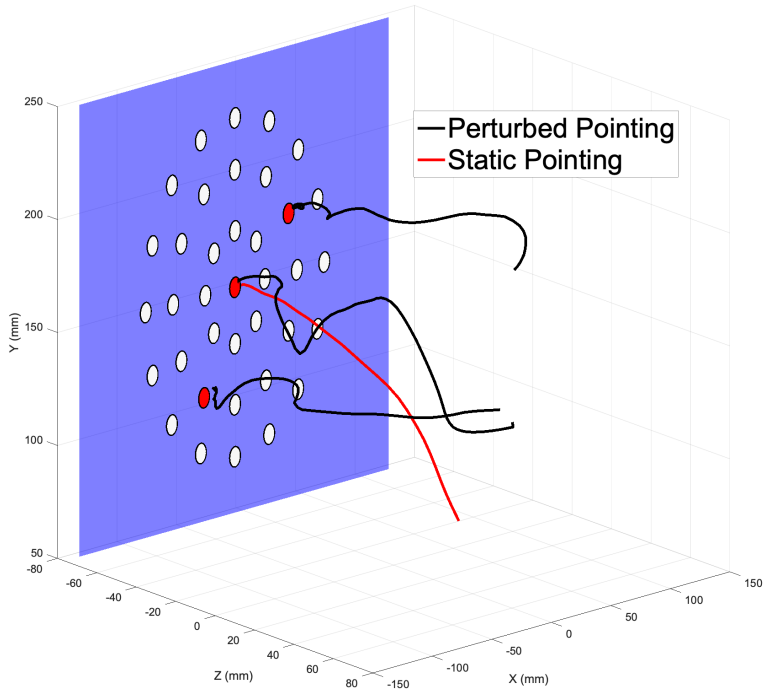


Fig. 3.4 Real hand pointing data

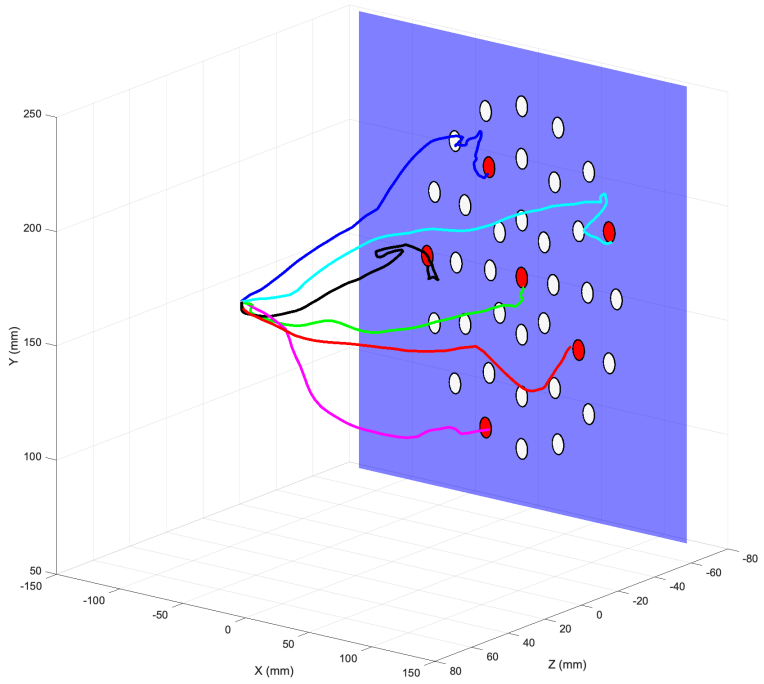


Fig. 3.5 Synthetic pointing data from the spatial ERV model; they are not used in intent inference performance evaluation.

improvement compared to the ERV model, and slightly outperforms the jump-ERV model in [69]. The lower success rate and larger standard deviation with pointing gestures in 3-D compared to those in Section 3.6.1 are due to the high number of nominal endpoints (37 instead of 16) with multi-circular layout. Readers are reminded that the chosen metric, as well as its limitations and alternative options, are elaborated upon in Section 2.5.0.1. Similar to previous intent inference tasks, the methods presented in this subsection significantly exceed the performance of a simple random guess, which would be around 2.7%. It's noteworthy that while the Lévy-ERV model doesn't outperform the BD-CA model in this task, as evidenced by its 56.79% average success rate in Fig. 2.6 from the previous chapter, it does exhibit a marked improvement over the Gaussian ERV model. The latter was demonstrated in [67] to surpass more traditional predictors, such as those based on nearest-neighbour methods.

3.6.3 Tracking a manoeuvring vessel

Lastly, we treat a conventional (sensor-level) tracking problem where the objective is to track a highly manoeuvring maritime target, as in [88], with the proposed stable

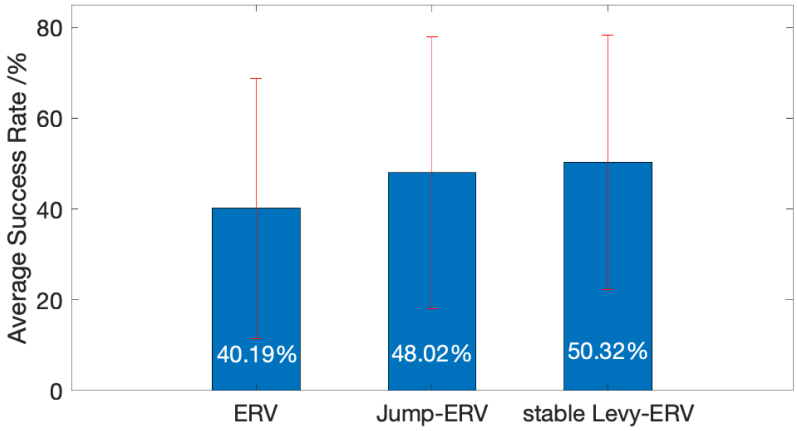


Fig. 3.6 Average intent inference success rate for freehand pointing movement in 3-D; red error bar is ±1 standard deviation.

Lévy model. The vessel trajectory [4] is depicted in Fig. 3.7. The voyage starts at time $t = 0$ and ends at $t = 2583$ such that the ground truth includes 118 position data points at irregular timesteps. To examine the presented Rao-Blackwillised particle filtering strategy, observations are obtained by adding Gaussian noise, with a standard deviation of 100, to the ground-truth data in each dimension. It should be noted that with such an observation model, the Gaussian Singer model is expected to track the vessel reasonably well. Thus, a significant improvement in the tracking performance with a stable Lévy model is not sought. Instead, below we compare the state estimation accuracy of the Gaussian Singer and stable Lévy Singer models, and outline the advantages of utilising the versatile parameters in the introduced stable Lévy modelling. We also use this real example to highlight key practical implementation aspects such as parameter sensitivity and computational requirements.

For evaluation purposes, we generate 100 measurement sets, each with $N = 118$ observations originating from the ground truth trajectory with an added Gaussian noise of covariance $R = 100^2 I_2$; a measurement set is displayed in Fig. 3.7. Here, the root mean square error (RMSE) is considered as the metric for tracking performance. It is defined, for one set, by

$$RMSE_{ds} = \frac{1}{N}\sqrt{\sum_{n=1}^{N} \|\hat{\boldsymbol{z}}_n - \boldsymbol{z}_n\|}, \tag{3.52}$$

[4]This dataset was kindly provided by QinetiQ Ltd., Winfrith.

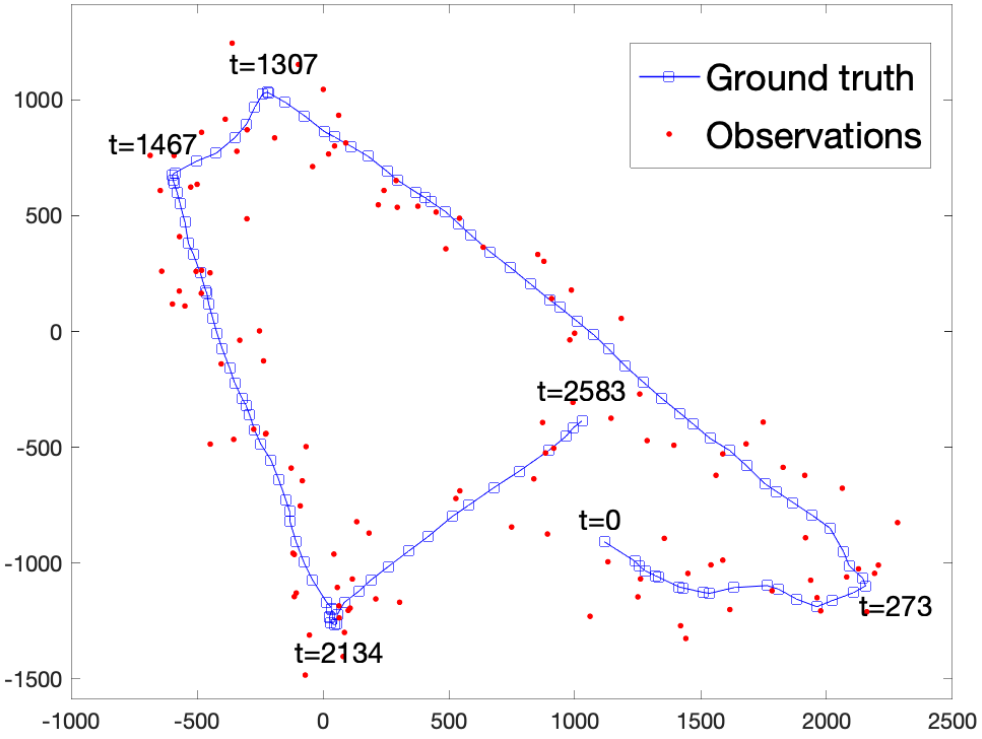


Fig. 3.7 Manoeuvring vessel truth trajectory from $t = 0$ to $t = 2583$ with 118 recorded locations and a measurement set.

where $\boldsymbol{z}_n$ is the ground truth of the vessel's position, $\hat{\boldsymbol{z}}_n$ is its estimated value, ds is the index of the measurement set (i.e. $ds = 1, 2, ..., 100$) and $N = 118$ is the total number of observations. The estimated position $\hat{\boldsymbol{z}}_n$ is extracted from the mean of the estimated state in (3.42) as per

$$\hat{\boldsymbol{z}}_n = \boldsymbol{C}\hat{\mathrm{E}}(\boldsymbol{x}_n|\boldsymbol{y}_{1:n}^{ds}) = \sum_{p=1}^{N_{\mathcal{P}}} \omega_n^{*(p,ds)} \boldsymbol{C}\boldsymbol{\mu}_{n|n}^{(p,ds)}, \tag{3.53}$$

where $\boldsymbol{C}$ is the mapping matrix that extracts the spatial position from the state vector $\boldsymbol{x}_n$ (in this experiment $\boldsymbol{C}$ is the observation matrix in (3.33)), and $\boldsymbol{\mu}_{n|n}^{(p,ds)}$ as well as $\omega_n^{*(p,ds)}$ are stored by the particle filter as in Algorithm 4. To quantify the overall tracking accuracy, we calculate the average RMSE over the total 100 measurement sets, dubbed M-RMSE, each with only one Monte Carlo run (e.g. for the Gaussian Singer model, it only involves running a Kalman-filter). To explore potential Monte Carlo implementation issues, we also define the P-RMSE as the average RMSE over 100 Monte Carlo runs with the same measurement set.

Table 3.3 Parameters and filtering performance of the stable Lévy (including Gaussian) Singer model. All models are with $\lambda = 10$, and for non-Gaussian ones $c = 0.1, N_{\mathcal{P}} = 5000$.

Value of α	M-RMSE(m) and tuned model noise
2 (Gaussian)	95.2517 ± 7.5061 ($\sigma_{Gaus} = 1.565$)
1.6	94.1214 ± 8.0208 ($\sigma_W = 2.013 \times 10^{-1}$)
1.3	93.3269 ± 8.2627 ($\sigma_W = 8.944 \times 10^{-2}$)
1.05	92.7565 ± 8.2672 ($\sigma_W = 3.578 \times 10^{-2}$)
0.9	92.4250 ± 8.4911 ($\sigma_W = 1.118 \times 10^{-2}$)
0.7	92.0316 ± 8.5049 ($\sigma_W = 2.236 \times 10^{-3}$)
0.5	91.7698 ± 8.5211 ($\sigma_W = 1.118 \times 10^{-4}$)
0.3	91.7146 ± 8.5943 ($\sigma_W = 1.342 \times 10^{-7}$)

In terms of the models parameterisation procedure, we first select the parameters for the well-known Gaussian Singer model that achieve a low M-RMSE. The model noise σ_{Gauss} for a fixed λ is then examined, noting that the tuned M-RMSE does not vary for $\lambda > 5$. Hence, we choose $\lambda = 10$ for the Gaussian Singer model, and commence tuning σ_W for the stable Lévy model with $\lambda = 10$. The M-RMSE for the stable Lévy Singer model and its standard deviation are listed in Table 3.3 for several values of the stability parameter α.

Table 3.3 shows that, for all presented α values, the proposed stable Lévy Singer models slightly outperform the Gaussian Singer model. Additionally, a noticeable trend is that the reduction of RMSE can be achieved with a lower α, which demonstrates the effectiveness of the α-stable extension of the conventional Gaussian noise in this simple linear Gaussian observation scenario. Here, we present only results for $\alpha > 0.3$ as our current implementations led to numerical issues[5] for lower values. Additionally, we note that we also experimented with a stable Lévy Langevin model for this dataset (model as described in Section 3.4.1.1 with a fixed $\lambda = 0.0001$) and this delivers similar performance to that of the analysed Lévy Singer model, and the M-RMSE was marginally improved, e.g. 93.1955 for $\alpha = 1.3$, and 91.4907 for $\alpha = 0.3$. For the Lévy Langevin model, lower α also resulted in a lower RMSE, when σ_W is manually tuned, following a similar trend to Table 3.3.

[5]In our dataset, the Lévy Singer model, when set with parameters $\lambda = 10$ and $\alpha < 0.3$, leads the predictive covariance matrix $\boldsymbol{P}_{n|n-1}^{(p)}$ in (3.37) for some particle p to have both extremely large and small diagonal elements. This can result in the Kalman filter update, as per the standard procedure in (3.37), not producing a positive semi-definite matrix $\boldsymbol{P}_{n|n}^{(p)}$ even when $\boldsymbol{P}_{n|n-1}^{(p)}$ is. Potential solutions include rescaling the covariance matrix, adopting the Joseph form of the Kalman filter update, or exploring other Kalman filter variations. For further insights, see [125].

Table 3.4 Run time T, filtering P-RMSE and M-RMSE of the stable Lévy Singer model for different values of Poisson series truncation c values and particle numbers $N_{\mathcal{P}}$.

c	$N_{\mathcal{P}}$	T (s)	P-RMSE (m)	M-RMSE (m)
0.6	500	19.90	106.77 ± 0.296	93.15 ± 8.18
0.1	5000	41.1785	106.69 ± 0.101	93.09 ± 8.25
0.1	1500	12.6012	106.70 ± 0.200	93.10 ± 8.25
0.1	500	4.2055	106.74 ± 0.344	93.19 ± 8.29
0.02	1500	4.9956	106.71 ± 0.202	93.14 ± 8.24
0.02	500	1.6633	106.73 ± 0.323	93.09 ± 8.22
0.003	500	1.1446	106.93 ± 0.344	93.43 ± 8.26
0.001	500	1.0929	108.11 ± 0.333	94.53 ± 8.19

We next utilise the same stable Lévy Singer model setup with $\lambda = 10$, $\alpha = 1.2$ and tuned model noise specified as $\sigma_W = 0.06708$ to investigate the tracking accuracy and the computational requirement (measured in terms of execution run time) for different truncation parameters c of the Poisson series representation (3.10) and number of particles $N_{\mathcal{P}}$. All algorithms are implemented in MATLAB, running on a laptop with Intel Core i9-9980HK at 2.4 GHz. The attained results are listed in Table 3.4 such that the P-RMSE is computed with the measurement set displayed in the Fig. 3.7 and 100 Monte Carlo runs are conducted.

It can be seen from Table 3.4 that a higher $N_{\mathcal{P}}$ can lower the standard deviation of the P-RMSE as it reduces the Monte Carlo error introduced by the particle filtering. It cannot, however, tangibly impact the standard deviation of the M-RMSE or the mean of both RMSE metrics. This is because the overall (average) tracking performance is sensitive to the measurement set, not to the filtering accuracy/uncertainty. Furthermore, the truncation parameter c only undermines the state estimation when its value is unreasonably low, e.g. $0.003, 0.001$. This is due to the additional Gaussianity (and less heavy-tailedness) induced by the inaccurate integral representation since a lower c incorporates fewer Poisson terms in (3.10) on page 62 with more residuals approximated by a Gaussian variable in (3.13) on page 63. A higher c or $N_{\mathcal{P}}$ inevitably requires more computational effort. However, it can be noticed that for a wide range of c and $N_{\mathcal{P}}$ values, the M-RMSE for $\alpha = 1.2$ in Table 3.4 is between that of $\alpha = 1.3$ and 1.05 in Table 3.3 as expected. Hence we can confidently anticipate that lower c and $N_{\mathcal{P}}$ values, for instance $c = 0.02$ and $N_{\mathcal{P}} = 500$ which only demands a run time of 1.66 seconds per measurement set, can still deliver a comparable performance as in Table 3.3.

We now illustrate the tracking performance by considering one particular set of observations, corresponding to a single generation of the observation noise process added

to the ship trajectory data. The stable Lévy Singer ($\alpha = 0.3$) and Gaussian Singer models with the parameters listed in Table 3.3 are employed to perform the smoothing tasks. For the Gaussian model, the Rauch–Tung–Striebel (RTS) smoother [36, 126] is applied to obtain the smoothed estimates. For the Lévy model, a straightforward smoothing strategy is carried out by running $N_{\mathcal{P}}$ RTS smoothers conditioned on the latent variable samples obtained from the particle filtering, i.e. the samples which provide empirical approximation in (3.39). The observations and smoothed trajectories for both models are depicted in Fig. 3.8, and the smoothing root squared error compared with the ground truth at each time-step is plotted in Fig. 3.9. It can be seen that the Lévy smoothed trajectory is improved compared to the tuned Gaussian model in two clear aspects: first, the Lévy model is better able to capture several of the sharp corners in the trajectory, see e.g. $t = 273$ where the Gaussian over-rounds the corner; and secondly, it is better able to capture the near straight-line trajectories in between the sharp manoeuvres, see e.g. times $t = 673$ to 1049. The improved tracking performance of the Lévy Singer model can also be noticed in Fig. 3.9, where the Lévy Singer model achieves a lower root squared error for most of the voyage (including during the two periods mentioned above).

To illustrate the potential of a Lévy state space model to capture abrupt manoeuvres compared to the standard Gaussian formulation, including estimating the velocity vector, we consider a synthetic 2-D trajectory with sudden changes. Both, the Lévy and Gaussian Singer models, are examined for the smoothing task. The track, displayed in Fig. 3.10, is generated by the stable Lévy Singer model with $\alpha = 1.2, \lambda = 100$, and $\sigma_W = 14.14$. A measurement set is then generated by adding a Gaussian noise with a covariance of $R = 0.2^2 I_2$. The ground-truth, measurements set and the estimated smoothed trajectories are shown in Fig. 3.10. The Lévy model uses the same parameters as in the synthetic track, whilst the Gaussian model is manually fine tuned to give the lowest average RMSE over 100 measurement sets generated with the same observation noise covariance (including the testing measurement set showed in Fig. 3.10), specifically $\lambda = 100$ and $\sigma_{Gaus} = 93.3381$.

Similar to the smoothing estimation for the real vessel data, it can be seen from Fig. 3.10 that the dynamic noise of a Gaussian model can be tuned to capture manoeuvres, despite still not following sharp turns and introducing over-smoothing effects (e.g. around $t = 9.15$ and $t = 11.55$). In contrast, the stable Lévy model offers more accurate tracking of abrupt turns and a lower overall RMSE (0.0809 and 0.0922 for the Lévy and Gaussian model respectively). To demonstrate its ability to capture sharp changes in the state, we examine the estimated velocity for the trajectory in

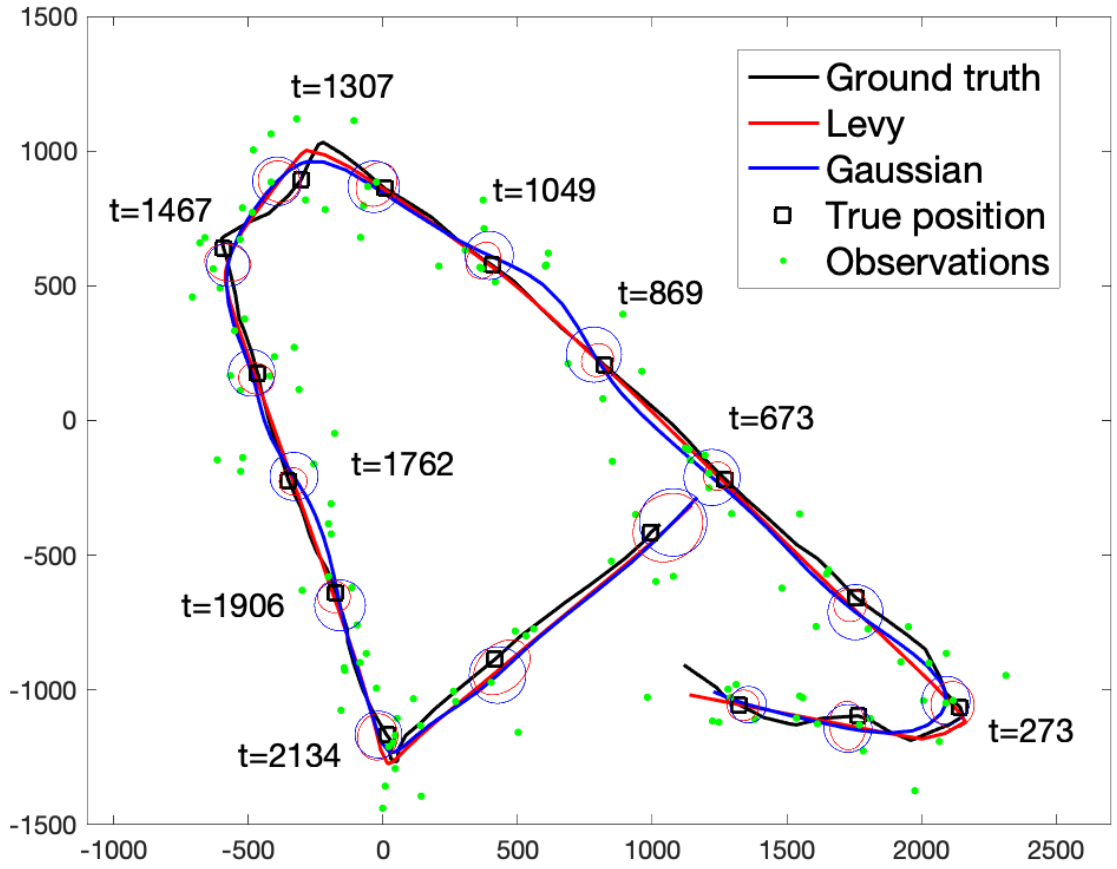


Fig. 3.8 Smoothed trajectories (mean of the estimate) from the Lévy ($\alpha = 0.3$) and Gaussian Singer models, and corresponding 95% confidence ellipse. The black squares are the true target positions corresponding to the error ellipse. Notably, the vessel makes four turns at times $t = 273, 1307, 1467$ and 2134.

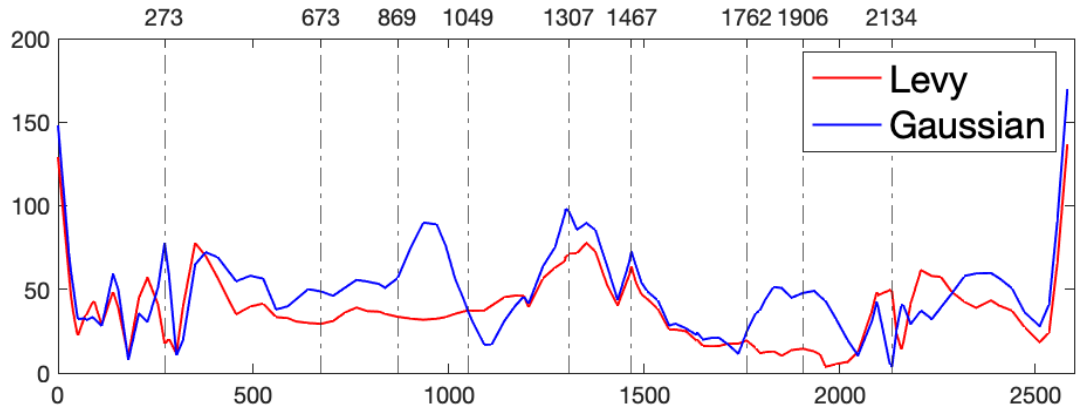


Fig. 3.9 Real time ($t = 0$ to 2583) root squared error compared with the ground truth for smoothing the observations in Fig. 3.8 with the Lévy ($\alpha = 0.3$) and Gaussian Singer models. The x-axis represents time t, while the y-axis indicates the root squared error value as defined in (3.52). Vertical bars mark the specific times showcased in Fig. 3.8.

Fig. 3.10. It can be clearly noticed from Fig. 3.11 that the stable Lévy Singer model can accurately follow the rapid changes in velocity (e.g. at $t = 8.3$, $t = 9.15$, and $t = 11.55$), whilst the Gaussian model fails to follow such sharp changes timely and instead over-smooths them. Furthermore, the Lévy model demonstrates a steady and accurate velocity estimation when the vessel exhibits a nearly constant velocity moving (e.g. from $t = 4$ to $t = 8$, from $t = 10$ to $t = 11.5$), and thus it provides useful information (e.g. direction of travel) for the position inference and intent inference tasks. For the Gaussian case, an increase in the dynamical noise is needed to (nearly) follow manoeuvres and reduce overall RMSE at the expense of higher noise during (nearly) straight trajectory sections where estimates are more exposed to noise in the available observations. Such an undesirable trade-off is not imposed by the proposed Lévy model,

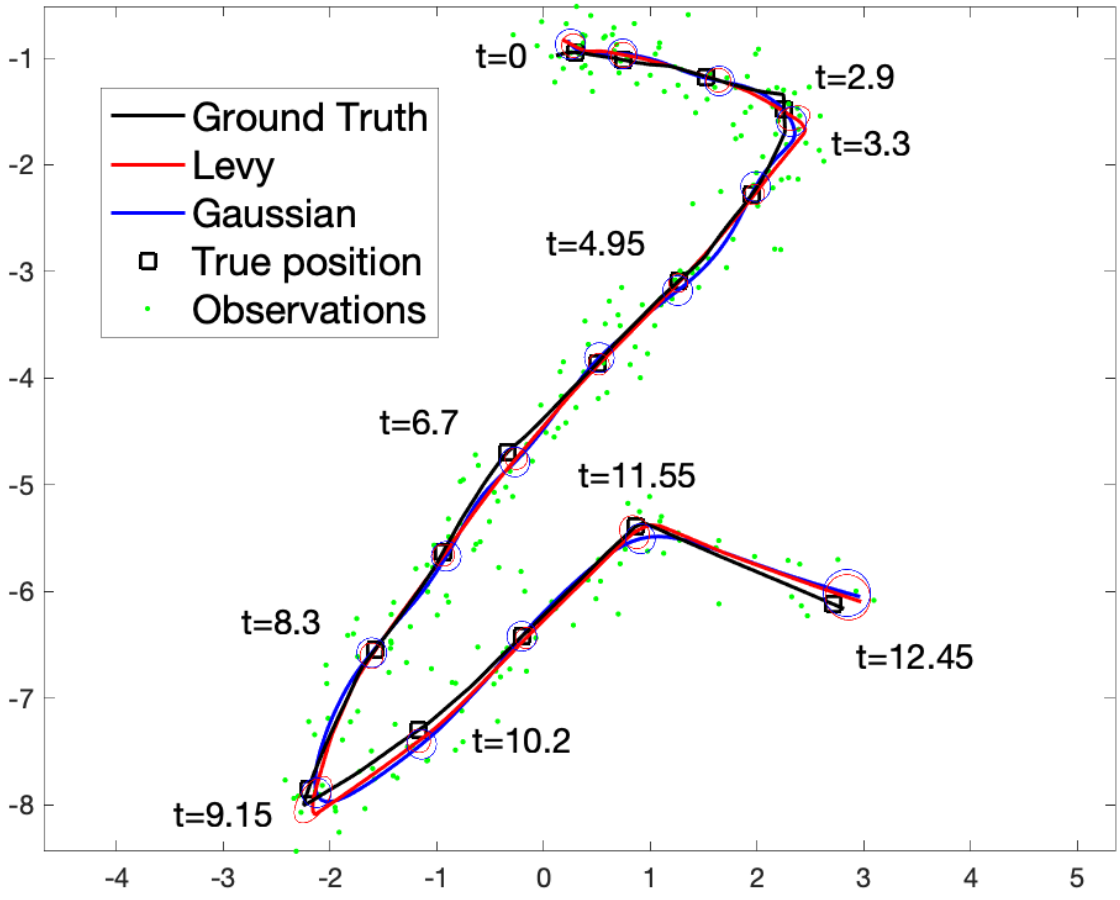


Fig. 3.10 Synthetic trajectories including 250 data points generated from the Lévy Singer ($\alpha = 1.2$) model and a measurement set. Smoothed trajectories with the Lévy and Gaussian Singer models are plotted with 95% confidence ellipse.

since its heavy-tailed behaviour can adapt automatically to both smooth/straight sections and to abrupt changes in state.

3.7 Conclusion

A new probabilistic framework, based on novel stable Lévy models, is proposed in this chapter for sensor-level (e.g. kinematic state estimation) and meta-level (e.g. destination prediction) tracking of highly manoeuvring objects. An efficient intent inference strategy with the latent destination model, which can apply to Gaussian and/or stable Lévy dynamic noise(s) is also introduced. Results from real data

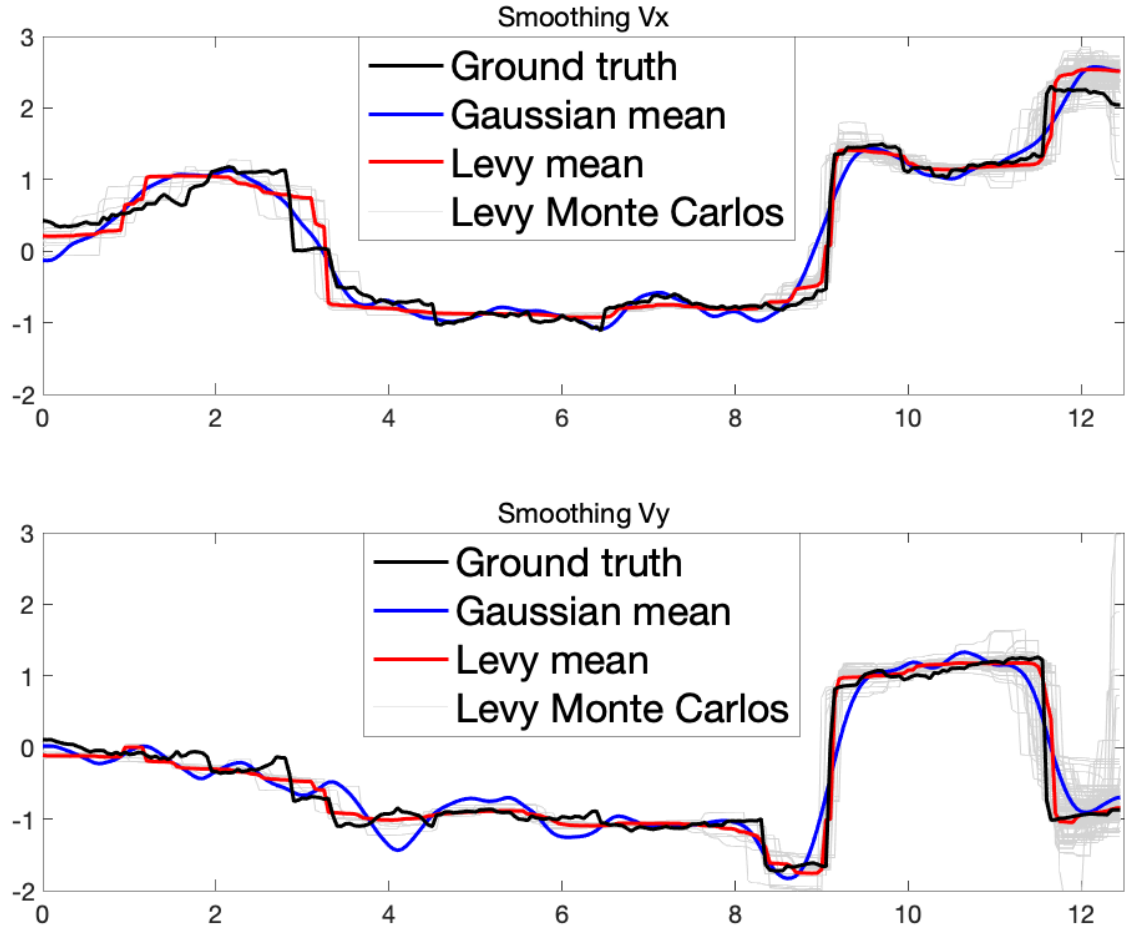


Fig. 3.11 Smoothed velocity estimates based on the observations in Fig. 3.10. Top and bottom panels are for x and y directions, respectively. The mean for 1) the Gaussian Singer model, 2) the Lévy Singer model, 3) each Monte Carlo conditioned Lévy Singer model estimate are also plotted.

illustrate the ability to deliver superior intent inference performance with the latent destination model, and demonstrate some useful performance improvements attained by the developed stable Lévy models compared with conventional Gaussian models.

From the classical sensor-level tracking example of a manoeuvring vessel and a synthetic highly manoeuvring trajectory, we can see that the proposed approach demonstrates its ability to track accurately the abrupt manoeuvres (e.g. sharp turns and/or accelerations). It also provides automatically a more steady kinematic state estimation on a near constant velocity movement. Although the overall positional estimates from a Lévy model are only slightly more accurate in our presented simulations, we anticipate that more significant tracking and intent improvements may be achieved in future work with other observation functions (e.g. non-linear and/or non-Gaussian), trajectories with more severe manoeuvres (such as flight patterns of drosophila from animal tracking where fruit flies can alter their heading by 90 degrees within approximately 50 ms [98]), and more general scenarios that may for example include clutter and data association issues (tracking scenarios involving clutter and data association will be discussed in Chapter 5 and Chapter 6). Other stable Lévy models, which have not been tested here (such as the constant velocity and constant acceleration models), may also provide performance enhancements for the general manoeuvring target tracking case. Whilst the stable Lévy model is formulated above as a continuous-time linear system driven by multivariate isotropic α-stable noise, a more general non-isotropic noise driven model can be explored in the future, see [21] for proposals in the univariate case.

Appendix 3.A Derivation of the variance of the residual terms

Consider the residual terms in (3.11) as the summation of Poisson series on the fixed interval $[c\Delta t, d\Delta t)$, with $d \to \infty$, and their number $N_{[c,d)}$ is defined by the Poisson distribution

$$N_{[c,d)} \sim \text{Poisson}\left((d-c)\Delta t\right), \tag{3.54}$$

and consequently the residuals terms can be expressed by

$$\boldsymbol{R}_c(\Delta t) = \lim_{d\to\infty} \Delta t^{1/\alpha} \sum_{i=1}^{N_{[c,d)}} \Gamma_i^{-1/\alpha} \mathbf{f}(\Delta t, V_i)\boldsymbol{U}_i, \tag{3.55}$$

with $\{\Gamma_i\}$ are now independently and uniformly distributed in the interval $[c\Delta t, d\Delta t)$. Additionally, the summands in (3.55) are independent and are zero mean. Subsequently, we have

$$\begin{aligned}
&\mathrm{Var}[\sum_{i=1}^{N_{[c,d)}} \Gamma_i^{-1/\alpha}\mathbf{f}(\Delta t, V_i)\boldsymbol{U}_i] \\
=&\mathrm{E}\left[\left(\sum_{i=1}^{N_{[c,d)}} \Gamma_i^{-1/\alpha}\mathbf{f}(\Delta t, V_i)\boldsymbol{U}_i\right)\left(\sum_{i=1}^{N_{[c,d)}} \Gamma_i^{-1/\alpha}\mathbf{f}(\Delta t, V_i)\boldsymbol{U}_i\right)^\top\right] \\
=&\mathrm{E}\left[\sum_{i=1}^{N_{[c,d)}} \Gamma_i^{-2/\alpha}\mathbf{f}(\Delta t, V_i)\boldsymbol{U}_i\boldsymbol{U}_i^\top\mathbf{f}(\Delta t, V_i)^\top\right] \\
&+\mathrm{E}\left[\sum_{i=1}^{N_{[c,d)}}\sum_{j\neq i, j=1}^{N_{[c,d)}} \Gamma_i^{-1/\alpha}\Gamma_j^{-1/\alpha}\mathbf{f}(\Delta t, V_i)\mathrm{E}[\boldsymbol{U}_i\boldsymbol{U}_j^\top]\mathbf{f}(\Delta t, V_j)^\top\right] \qquad (3.56)\\
=&\mathrm{E}\left[\sum_{i=1}^{N_{[c,d)}} \Gamma_i^{-2/\alpha}\mathbf{f}(\Delta t, V_i)\boldsymbol{U}_i\boldsymbol{U}_i^\top\mathbf{f}(\Delta t, V_i)^\top\right] \qquad (3.57)\\
=&\mathrm{E}[N_{[c,d)}]\mathrm{E}[\Gamma_i^{-2/\alpha}]\mathrm{E}[\mathbf{f}(\Delta t, V_i)\boldsymbol{U}_i\boldsymbol{U}_i^\top\mathbf{f}(\Delta t, V_i)^\top],
\end{aligned}$$

where the last equality arises from the independence of each summand in (3.57), with both $\Gamma_i^{-2/\alpha}$ and $\mathbf{f}(\Delta t, V_i)\boldsymbol{U}_i\boldsymbol{U}_i^\top\mathbf{f}(\Delta t, V_i)^\top$ also being independent. The expectation in (3.56) is zero since all $\mathrm{E}[\boldsymbol{U}_i\boldsymbol{U}_j^\top]$ $(i \neq j)$ is zero, given that $\boldsymbol{U}_i$ and $\boldsymbol{U}_j$ are independent with zero means. Recall that $N_{[c,d)}$ is Poisson distributed with rate $(d-c)\Delta t$, Γ_i is uniformly distributed in $[c\Delta t, d\Delta t)$, V_i is uniformly distributed between $(0, \Delta t]$, and $\boldsymbol{U}_i \sim \mathcal{N}(\mathbf{0}, \sigma_W^2 I_m)$, we have

$$\begin{aligned}
\mathrm{E}[N_{[c,d)}] &= (d-c)\Delta t, \\
\mathrm{E}[\Gamma_i^{-2/\alpha}] &= \frac{1}{(d-c)\Delta t}\int_{c\Delta t}^{d\Delta t} \Gamma_i^{-2/\alpha} d\Gamma_i \\
&= \frac{d^{1-2/\alpha} - c^{1-2/\alpha}}{d-c}\Delta t^{-2/\alpha}\frac{\alpha}{\alpha-2},
\end{aligned}$$

and

$$\begin{aligned}
&\mathrm{E}[\mathbf{f}(\Delta t, V_i)\boldsymbol{U}_i\boldsymbol{U}_i^\top\mathbf{f}(\Delta t, V_i)^\top] \\
=&\mathrm{E}_{V_i}[\mathbf{f}(\Delta t, V_i)\mathrm{E}_{U_i}[\boldsymbol{U}_i\boldsymbol{U}_i^\top]\mathbf{f}(\Delta t, V_i)^\top] \\
=&\frac{\sigma_W^2}{\Delta t}\int_0^{\Delta t} e^{\boldsymbol{A}(\Delta t-u)}\boldsymbol{H}\boldsymbol{H}^\top e^{\boldsymbol{A}(\Delta t-u)^\top} du = \frac{\sigma_W^2}{\Delta t}\boldsymbol{Q}(\Delta t),
\end{aligned}$$

Therefore, the sought variance reduces to

$$\begin{aligned}\mathrm{Var}[\boldsymbol{R}_c(\Delta t)] &= \lim_{d\to\infty} \Delta t^{2/\alpha}\mathrm{Var}[\sum_{i=1}^{N_{[c,d)}} \Gamma_i^{-1/\alpha}\mathbf{f}(\Delta t, V_i)\boldsymbol{U}_i] \\ &= \lim_{d\to\infty} \left(d^{1-2/\alpha} - c^{1-2/\alpha}\right) \sigma_W^2 \frac{\alpha}{\alpha-2}\boldsymbol{Q}(\Delta t) \\ &= \sigma_W^2 \frac{\alpha}{2-\alpha} c^{1-2/\alpha}\boldsymbol{Q}(\Delta t).\end{aligned}$$

Appendix 3.B Derivation of the matrix fraction decomposition

The integral in (3.15) can be expressed as

$$\boldsymbol{M}(\Delta t) = \int_0^{\Delta t} e^{\boldsymbol{A}(\Delta t-u)}\boldsymbol{B}du = \int_0^{\Delta t} e^{\boldsymbol{A}(\Delta t-u)}du\boldsymbol{B}.$$

Let $\mathcal{J}(\Delta t) = \int_0^{\Delta t} e^{\boldsymbol{A}(\Delta t-u)}du$, and the aim is to prove that $\mathcal{J}(\Delta t)$ is equal to $\boldsymbol{J}_M$ in (3.25). By *Leibniz integral rule*, we have

$$\begin{aligned}&\frac{d}{d\Delta t}\mathcal{J}(\Delta t) = \frac{d}{d\Delta t}\int_0^{\Delta t} e^{\boldsymbol{A}(\Delta t-u)}du \\ &= \int_0^{\Delta t} \frac{d}{d\Delta t}e^{\boldsymbol{A}(\Delta t-u)}du + e^{\boldsymbol{A}(\Delta t-\Delta t)} = \boldsymbol{A}\mathcal{J}(\Delta t) + I_{s_A},\end{aligned}$$

recalling that $\boldsymbol{A}$ and the identity matrix I_{s_A} are of size $s_A \times s_A$. We can then construct the linear differential equation

$$\frac{d}{d\Delta t}\begin{bmatrix}\mathcal{J}(\Delta t) \\ I_{s_A}\end{bmatrix} = \begin{bmatrix}\boldsymbol{A} & I_{s_A} \\ \mathbf{0}_{s_A} & \mathbf{0}_{s_A}\end{bmatrix}\begin{bmatrix}\mathcal{J}(\Delta t) \\ I_{s_A}\end{bmatrix}. \tag{3.58}$$

It can be easily seen that equation (3.58) has the solution

$$\begin{aligned}\begin{bmatrix}\mathcal{J}(\Delta t) \\ I_{s_A}\end{bmatrix} &= \mathrm{expm}\left(\begin{bmatrix}\boldsymbol{A} & I_{s_A} \\ \mathbf{0}_{s_A} & \mathbf{0}_{s_A}\end{bmatrix}\Delta \mathrm{t}\right)\begin{bmatrix}\mathcal{J}(0) \\ I_{s_A}\end{bmatrix} \\ &= \mathrm{expm}\left(\begin{bmatrix}\boldsymbol{A} & I_{s_A} \\ \mathbf{0}_{s_A} & \mathbf{0}_{s_A}\end{bmatrix}\Delta \mathrm{t}\right)\begin{bmatrix}\mathbf{0}_{s_A} \\ I_{s_A}\end{bmatrix},\end{aligned} \tag{3.59}$$

which is exactly (3.25) so that we have $\mathcal{J}(\Delta t) = \boldsymbol{J}_M$.

Chapter 4

Conditionally Factorised Variational Bayes with Importance Sampling

The approximate inference methods employed in previous chapters are limited to the Monte Carlo methods, which were verified to deliver satisfactory results for the considered inference problem. However, the Monte Carlo methods can be computationally demanding, and the requirement for real-time processing often limits their application in more complex probabilistic inference settings, such as multi-object tracking with clutter. This motivates us to consider other inference methods with faster implementation. Subsequently, an efficient approximate inference technique named coordinate ascent variational inference (CAVI) will be the main ingredient of our inference paradigm for the remaining chapters of this thesis.

CAVI is a popular approximate inference method; however, it relies on a mean-field assumption that can lead to large estimation errors for highly correlated variables. In this chapter, we propose a conditionally factorised variational family with an adjustable conditional structure to retain the dependence between desired variables, and derive the corresponding coordinate ascent algorithm for optimisation. The algorithm is termed Conditionally factorised Variational Bayes (CVB) and implemented with importance sampling. We show that by choosing a finer conditional structure, our algorithm can be guaranteed to achieve a better variational lower bound, thus providing a flexible trade-off between computational cost and inference accuracy. The validity of the method is demonstrated in a simple posterior computation task. Some of the results from this chapter have been published in [127]. [1].

[1] © 2022 IEEE. Reprinted, with permission, from [127]

4.1 Introduction

As one of the earliest formulations of variational inference [23, 28], CAVI (also known as mean-field variational inference [128, 12]) is a popular method for approximation of intractable posterior densities for Bayesian models. Compared to another well-known Bayesian approximate inference method Markov chain Monte Carlo (MCMC), CAVI is faster, easier to monitor its convergence, and can typically yield a comparable inference accuracy. Moreover, the approximate posterior obtained by CAVI can be efficiently stored by a small number of parameters if its parametric form is well-known, in contrast to the large number of samples required to represent the approximate posterior in the Monte Carlo method. For these reasons, CAVI has been widely applied as an alternative strategy to MCMC. For example, in the signal processing field, CAVI has been employed to develop approximate Bayesian filters [129, 130], smoothers [128, 131], and to carry out approximate Bayesian parameter estimation for state space models [128, 59].

Despite the above-mentioned benefits, CAVI relies on the mean-field assumption: it assumes independent fully factorised variational posteriors. Such an assumption renders the approximation inaccurate for highly correlated variables and may introduce additional local optima [132, 28]. To mitigate this problem, this chapter presents the CVB, a coordinate ascent optimiser that can theoretically yield a better approximation with an adjustable conditional variational distribution preserving the dependence between variables.

4.1.1 Problem formulation

Consider a Bayesian model, where the set of all observed variables is denoted as Y, and let X denote the set of parameters and latent variables whose posterior $p(X|Y)$ is of interest but intractable. This chapter focuses on approximating this posterior within the variational inference framework. To this end, we first propose a family of variational distributions q, and then find the member of this family which minimises the Kullback-Leibler (KL) divergence to the exact posterior. Such a procedure is equivalent to finding

$$q^* = \arg\max_q \mathcal{F}(q), \tag{4.1}$$

subject to the restriction that q belongs to the predefined family. The evidence lower bound (ELBO) $\mathcal{F}(q)$ in (4.1) is defined as

$$\mathcal{F}(q) = \mathrm{E}_{q(X)} \log \frac{p(X,Y)}{q(X)}. \tag{4.2}$$

In this chapter, we propose to use a conditionally factorised family for the variational distribution q — a generic and flexible family where the dependence between the desired variables can be constructed with user selected detail. It encompasses the standard mean-field family as a special case.

4.1.2 Background of CAVI

Classical CAVI is a variational inference algorithm where the variational distribution q belongs to a mean-field family q_{mf}. This family assumes that the approximated distribution $q_{mf}(X)$ can be independently factorised as follows,

$$q_{mf}(X) = \prod_{i=1}^{\nu} q_{mf}^{i}(x_i), \tag{4.3}$$

where each $x_i (i = 1, 2, ..., \nu)$ is disjointly partitioned from X, i.e. $X = \{x_1, x_2, ..., x_\nu\}$, and each q_{mf}^{i} is a *free-form* distribution (a variational distribution q_{mf}^{i} is *free-form* if there is no predefined parametric form for it). With such an assumption in (4.3), the optimisation problem in (4.1) can be solved by a coordinate ascent algorithm. Specifically, for $i = 1, 2, ..., \nu$, the algorithm iteratively updates $q_{mf}^{i}(x_i)$ by the optimisation in (4.4) while keeping q_{mf}^{i-} fixed, where $q_{mf}^{i-} = \prod_{l \neq i} q_{mf}^{l}(x_l)$.

$$\arg\max_{q_{mf}^{i}} \mathcal{F}(q_{mf}) \propto \exp(\mathrm{E}_{q_{mf}^{i-}(x_{i-})} \log p(X,Y)) \tag{4.4}$$

This update is a fundamental result based on variational calculus, and the derivation can be found in [23, 12, 128]. As the name CAVI suggests, each update step in (4.4) is coordinate-wise and guarantees a non-negative increment of the ELBO, such that (4.2) will eventually reach a local maximum of the optimisation problem (4.1).

In some cases, the optimal distribution in (4.4) may not be analytically tractable. A remedy to this issue is to sample from intractable variational distributions, thereby using Monte Carlo methods to approximate the other optimal distributions in closed-form expressions. For example, to carry out variational inference for models with constrained prior, [133] uses MCMC to obtain samples for the intractable update in the standard CAVI procedure; and, [59] applies sequential Monte Carlo within CAVI

procedure to estimate the parameters of a nonlinear state space model. Approximating intractable coordinate ascent updates with Monte Carlo methods is also a key concept utilised in the implementation of the proposed CVB algorithm.

4.1.3 Related work

Compared to CAVI (Section 4.1.2), modern gradient-based variational inference algorithms such as stochastic variational inference [29], black-box variational inference [30] and variational auto-encoders [31] typically require a predefined parametric form for the variational distribution, and they can be easily scaled to large datasets and are sometimes faster and easier to implement in modern machine learning tasks. However, to our knowledge, the procedure of computing the required gradient for these algorithms can be laborious and even render it inapplicable for complex models which may occur in the signal processing field, e.g. models subject to hard constraints [133], and/or models involving complicated compositional variables such as Poisson process arrival times [94, 69]. In the contrast, the variational distributions arising from coordinate ascent updates such as (4.4) can naturally account for the complex model priors. This makes CAVI applicable to a wider range of probabilistic models. For example, if the prior $p(x^i)$ is a truncated distribution, then the optimal distribution in (4.4), even approximated with Monte Carlo methods, still preserves the same truncation (see [133]). For this reason, in this thesis we focus on the variational inference method which only requires *free-form* coordinate ascent update for its flexibility in a wide range of probabilistic models.

Restoring the dependence within the variational distributions has been studied since the early days of the variational inference method [134]. This is known as structured variational inference in the machine learning community, see [28, 23, 132]. Early research on structured variational inference is often model-specific and limited to the graphical model setting [134, 135]. Attempts to improve the mean-field approximation in a general probabilistic model, to our knowledge, began with [136], whose parameters are numerically optimised to increase the ELBO. More recently, the conditional variational distributions employed in [132, 137] are more closely related to our work; the conditional structure they implicitly assume (termed *conditional everywhere* in this thesis) can be regarded as a limiting case of the *setwise conditional* structure, which we propose and define in this chapter for the conditionally factorised family. Moreover, whilst [132, 137] focus on deriving the update for stochastic variational inference , we consider deriving the coordinate-ascent update for *free-form* variational distributions.

4.1.4 Contributions

The first major contribution in this chapter is the introduction of a conditionally factorised variational family that can flexibly account for the conditional structure between variables, encompassing the standard mean-field family, e.g. in [23] as a special case. Besides mean-field structure, the conditional structures assumed in prior works [132, 137] can also be viewed as specific cases of the conditional structure proposed within our conditionally factorised variational family. Moreover, this chapter derives the resulting theoretical coordinate ascent updates in detail and incorporate them into the proposed theoretical CVB algorithm. Furthermore, this chapter shows that a finer conditional structure in CVB can be guaranteed to achieve a better ELBO, thus providing a flexible trade-off between computational cost and inference accuracy.

Nonetheless, implementing the theoretical CVB is generally intractable. As my second major contribution, this chapter proposes an importance sampling-based CVB that approximates the intractable coordinate ascent update within the importance sampling framework. The applicability of this proposed algorithm is discussed and it is shown that the algorithm yields an estimated ELBO as a byproduct. I prove that a finer conditional structure also ensures a higher estimated ELBO, a property analogous to that of the theoretical CVB. The guaranteed performance improvement over the standard mean-field CAVI is showcased through a simple example.

4.1.5 Layout

The remaining sections in the chapter are organised as follows. Section 4.2 introduces the conditionally factorised variational family and the theoretical CVB. Section 4.3 presents the importance sampling-based CVB and discusses its applicability and relevant properties. Section 4.4 provides detailed proofs and derivations for the theorems and approximations introduced in the previous two sections. Section 4.5 validates the proposed method, and Section 4.6 concludes the chapter.

4.2 Conditionally factorised variational Bayes

Consider the problem formulated in Section 4.1.1. Assume X includes at least two variables. We partition it into two disjoint parts, denoted by the global variable S and local variable Z (i.e. $X = \{S, Z\}$). If required, the local variable Z can be further partitioned into N_c ($N_c \geq 1$) variables, i.e. $Z = \{Z_1, Z_2, ..., Z_{N_c}\}$. Note that we do not impose any restriction on the type of the global or local variables: they can be

continuous, discrete or mixed type. The variational distribution q and exact distribution p used in this chapter should be understood as the probability mass function (PMF) for discrete variables; and as the generalised probability density function (PDF) for continuous variables and mixed-type variables, where the Dirac delta function is applied to the discrete part of the variable domain.

4.2.1 Conditionally factorised variational family

A member of the proposed conditionally factorised family should first satisfy the following factorisation:

$$q(X) = q_g(S) \prod_{i=1}^{N_c} q_c^i(Z_i|S). \tag{4.5}$$

Note that our CVB method also applies to $N_c = 1$, whereas a factorisation may improve the tractability of variational distributions.

Denote the set of all factorised variational distributions in (4.5) by $\{q\}_{cf} = \{q_g, q_c^1, ..., q_c^{N_c}\}$. No predefined relationship is assumed between distributions in $\{q\}_{cf}$ and thus coordinate-wise update for optimisation in (4.1) may be carried out, i.e. one of the distributions in $\{q\}_{cf}$ is updated whilst the others are kept fixed. Furthermore, the global distribution q_g is a regular *free-form* variational distribution in a similar sense to the *free-form* distribution in (4.3). Each conditional variational distribution q_c^i is a function of both S and Z_i defined as follows. Denote the domain of S as U (i.e. $S \in U$).

Definition 4.2.1. $q_c^i(Z_i|S)$ is *setwise conditional* on a partition $\mathcal{P}_i$ if 1) $\mathcal{P}_i$ is a partition of U; and 2) the distribution $q_c^i(Z_i|S)$ can be written as

$$q_c^i(Z_i|S) = \sum_{A \in \mathcal{P}_i} q_c^{i,A}(Z_i) I(S \in A), \tag{4.6}$$

where $I(S \in A)$ is the indicator function; $\{q_c^{i,A}(Z_i) : A \in \mathcal{P}_i\}$ consists of $|\mathcal{P}_i|$ *free-form* variational distributions of Z_i which have no predefined relationship with each other.

Remark 1. In this thesis, we adopt the definition of a partition of a set from [138], i.e. a partition of U is a set of non-empty sets which cover U without overlap. Specifically, $\mathcal{P}_i$ is a partition of U if 1) for all $A \in \mathcal{P}_i$, $A \neq \emptyset$; and 2) for all $A_1, A_2 \in \mathcal{P}_i$, $A_1 \cap A_2 = \emptyset$; and 3) $\bigcup_{A \in \mathcal{P}_i} A = U$.

Remark 2. A special case of *setwise conditional* is when the partition $\mathcal{P}_i$ is the trivial partition, i.e. $\mathcal{P}_i = \{U\}$, when q_c^i no longer depends on S and the proposed conditionally

factorised assumption in (4.5) degenerates to the standard mean-field assumption in (4.3) if $\mathcal{P}_i = \{U\}$ for all $i = 1, 2, ..., N_c$. Moreover, a limiting case of *setwise conditional* can be achieved by setting the partition $\mathcal{P}_i$ to the partition of singleton, i.e. $\mathcal{P}_i = \{\{a\} : a \in U\}$. In this case $|\mathcal{P}_i| = \infty$ if S includes a continuous variable. This limiting case is termed as *conditional everywhere*, whose formal definition can be found below.

Definition 4.2.2. $q_c^i(Z_i|S)$ is *conditional everywhere* if 1) for any $k \in U$, $q_c^i(Z_i|S = k)$ is a *free-form* variational distributions of Z_i; and 2) for any $j \in U$ and $j \neq k$, there is no predefined relationship between $q_c^i(Z_i|S = j)$ and $q_c^i(Z_i|S = k)$.

We now complete the definition of the conditionally factorised family. Each a realisation of $\{q\}_{cf}$ that satisfies the above definition is a member of the conditionally factorised family. Given the factorised variables $\{S, Z_1, ..., Z_{N_c}\}$, the proposed conditionally factorised family is only parameterised by the partitions introduced in the Definition 4.2.1, i.e. $\{\mathcal{P}_i : i = 1, 2, ..., N_c\}$.

It is informative to consider how the conditionally factorised family is affected by the associated partitions. Specifically, it can be easily verified that for variables $\{S, Z_1, ..., Z_{N_c}\}$, one conditionally factorised family is *wider* than another (each member of the latter is a member of the former) if each associated partition $\mathcal{P}_i$ $(i = 1, 2, ..., N_c)$ of the former family is *finer* than the corresponding partition of the latter (every set in the former partition is a subset of a set in the latter partition). Therefore, the conditionally factorised family with *conditional everywhere* q_c^i for all $i = 1, 2, ...N_c$ is the *widest* family of the proposed framework, since each q_c^i has the *finest* partition. Also, the proposed conditionally factorised family is always *wider* than the fully factorised mean-field family for variables $\{S, Z_1, ..., Z_{N_c}\}$, since the latter can be considered as a special case of the conditionally factorised family that assumes the trivial partitions for all q_c^i.

4.2.2 Coordinate ascent update

The aim of variational inference is to find the member of the proposed conditionally factorised family that solves the optimisation problem in (4.1). A *wider* family guarantees a better global optimum, but it may take more effort to search for it. Here a coordinate ascent algorithm is sought for the optimisation. Specifically, the parametric form of each distribution from $\{q\}_{cf}$ is iteratively updated (whilst other variational distributions are kept fixed) by the optimal distribution given in the following theorem.

Theorem 4.2.1. *Denote the optimal global distribution by* $\tilde{q}_g(S) = \arg\max_{q_g} \mathcal{F}(\{q\}_{cf})$*, and the optimal* $q_c^i(Z_i|S)$ *by* $\tilde{q}_c^i(Z_i|S) = \arg\max_{q_c^i} \mathcal{F}(\{q\}_{cf})$ *for* $i = 1, 2, ..., N_c$*, then we have the unique* $\tilde{q}_g(S)$*, i.e.*

$$\tilde{q}_g(S) \propto \frac{\exp\left(\mathrm{E}_{\prod_{i=1}^{N_c} q_c^i(Z_i|S)} \log p(X,Y)\right)}{\prod_{i=1}^{N_c} \exp\left(\mathrm{E}_{q_c^i(Z_i|S)} \log q_c^i(Z_i|S)\right)}. \tag{4.7}$$

For each q_c^i *which is setwise conditional on a partition* $\mathcal{P}_i$*, we have*

$$\tilde{q}_c^i(Z_i|S) = \sum_{A \in \mathcal{P}_i} \tilde{q}_c^{i,A}(Z_i) I(S \in A), \tag{4.8}$$

where each $\tilde{q}_c^{i,A}(Z_i)$ *is defined as follows: for* $A \in \mathcal{P}_i$ *such that* $q_g(S) = 0$ *for all* $S \in A$*,* $\tilde{q}_c^{i,A}(Z_i)$ *can be any distribution; otherwise* $\tilde{q}_c^{i,A}(Z_i)$ *is*

$$\tilde{q}_c^{i,A}(Z_i) \propto \exp\left(\frac{\int_A q_g(S) \mathrm{E}_{q_c^{i-}(Z_{i-}|S)} \log p(X,Y) dS}{\int_A q_g(S) dS}\right). \tag{4.9}$$

For each q_c^i *which is conditional everywhere,* $\tilde{q}_c^i(Z_i|S = u)$ *can be any distribution for all* u *such that* $q_g(S = u) = 0$*; for other* u *such that* $q_g(S = u) \neq 0$*, we have*

$$\tilde{q}_c^i(Z_i|S = u) \propto \exp\left(\mathrm{E}_{q_c^{i-}(Z_{i-}|S=u)} \log p(Z, Y, S = u)\right). \tag{4.10}$$

Remark 1. The integral in (4.9) is in a general sense: it can be replaced by a summation if S is discrete variable or q_g is a PMF. The $q_c^{i-}(Z_{i-}|S)$ in (4.9) and (4.10) denotes $\prod_{l \neq i} q_c^l(Z_l|S)$ for $N_c > 1$; and when $N_c = 1$, the operator $\mathrm{E}_{q_c^{i-}}$ should be neglected, i.e. $\mathrm{E}_{q_c^{i-}(Z_{i-}|S)}[\cdot] = [\cdot]$.

Remark 2. In general, the optimisers in (4.9) and (4.10) are not necessarily unique. However, we note that for $A \in \mathcal{P}_i$ such that the non-negative integral $\int_A q_g(S) dS$ is neither zero or infinitesimal, we have the unique $\tilde{q}_c^{i,A}$ in (4.9). More details about the uniqueness of the optimiser are discussed at the end of Section 4.4.1.2.

Remark 3. Since *setwise conditional* is a general representation for q_c^i, its corresponding update (4.9) is also a general formula. Specifically, the *conditonal everywhere* update in (4.10) and the standard mean-field update in (4.4) can be recovered from (4.9) by setting $\mathcal{P}_i$ to the corresponding partitions discussed in the Remark 2 for Definition 4.2.1.

The laborious proof of Theorem 4.2.1 is presented in Section 4.4.1 on page 116.

For any assumed factorised variables $\{S, Z_1, ..., Z_{N_c}\}$, the theoretical coordinate ascent algorithm, denoted as theoretical CVB, is summarised in Algorithm 7. As with the standard CAVI, the ELBO is guaranteed to monotonically increase over iterations of the theoretical CVB. This is due to: 1) each update in Algorithm 7 is guaranteed to increase the ELBO since they are derived from the exact $\arg\max \mathcal{F}(\{q\}_{cf})$ operator, and 2) the refinement step can keep the current ELBO since the resulting *wider* conditionally factorised family includes the latest updated parametric form of $\{q\}_{cf}$ as a member (as discussed above), and allows the ELBO to be optimised in a *wider* variational family. When all the partitions $\mathcal{P}_i, i = 1, 2, ..., N_c$ are fixed, the theoretical CVB will converge to a local optimum of the problem in (4.1).

Algorithm 7: Theoretical CVB

while *the ELBO* $\mathcal{F}(\{q\}_{cf})$ *not converged* **do**
  Update $q_g \leftarrow \tilde{q}_g$ according to (4.7).
  for $i = 1 : N_c$ **do**
    Refine the $\mathcal{P}_i$ if higher accuracy is required.
    foreach $A \in \mathcal{P}_i$ **do**
      if *the density* $q_g(S) = 0$ *for all* $S \in A$ **then**
        Set $q_c^{i,A}$, or if A is a singleton set, then set $q_c^i(Z_i|S = u)$ where $u \in A$, to an arbitrary distribution.
      else
        if *A is a singleton set* **then**
          Update $q_c^i(Z_i|S = u) \leftarrow \tilde{q}_c^i(Z_i|S = u)$ where $u \in A$, via (4.10).
        else
          Update $q_c^{i,A} \leftarrow \tilde{q}_c^{i,A}$ via (4.9).
        end
      end
    end
  end
end

We can now see why the theoretical CVB can be guaranteed to improve the performance of the standard CAVI. As the standard CAVI can be considered as the CVB with the trivial partitions, when such a CAVI has converged, we can always refine the partitions within the proposed CVB framework and the ELBO is guaranteed to monotonically increase again. Therefore, the theoretical CVB with *conditionally everywhere* q_c^i which assumes the *finest* partition for all $i = 1, 2, ..., N_c$ is the most accurate setting of the proposed conditionally factorised family; however, it comes with the most intensive computations as there can be an infinite number of *free-form*

distributions (i.e. $q_c^i(Z|S=u)$ for all $u \in U$) to be optimised. On the other hand, the *setwise conditional* q_c^i offers a flexible trade-off between accuracy and computational efficiency, i.e. a *finer* partition $\mathcal{P}_i$ can always yield a higher accuracy with more *free-form* distributions (i.e. $q_c^{i,A}$ for $A \in \mathcal{P}_i$) to be optimised.

4.3 Importance sampling based CVB

The implementation of the theoretical CVB (i.e. Algorithm 7) is highly intractable except for special cases (e.g. the global variable set S only includes a discrete variable with finite range of values). The intractability is two-fold, 1) the optimal distribution $\tilde{q}_g$ in (4.7) and $\tilde{q}_c^{i,A}$ in (4.9) can rarely be evaluated analytically, and 2) it is computationally prohibitive to evaluate all $\tilde{q}_c^{i,A}$ when the partition $\mathcal{P}_i$ has too many elements (note that $|\mathcal{P}_i| = \infty$ for *conditional everywhere* q_c^i). Therefore, we will use a Monte Carlo method to approximate the intractable updates in the theoretical CVB, specifically, we will attempt to sample the global variable S from the optimal $\tilde{q}_g$ in (4.7), and these samples will then be employed to approximate the optimal update for q_c^i. As will be shown later, such a strategy only requires evaluating $q_c^i(Z_i|S)$ for some particular S, and can often lead to a closed-form approximation for $\tilde{q}_c^{i,A}$ in (4.9) (discussed in Section 4.3.3). Here the importance sampling technique is employed to carry out the required Monte Carlo approximation due to its relative efficiency compared to MCMC, and its ability to estimate the ELBO (as will be demonstrated shortly).

4.3.1 Algorithm

We assume S is a low-dimensional variable such that a standard importance sampling can effectively carry out the sampling. Note that this framework can be extended to incorporate a high dimensional S in a sequential Monte Carlo scheme, and this case will be presented in future. Suppose we have N_p particles sampled independently from the proposal $\lambda(S)$. We define the particle index set by $\mathcal{I} = \{1, 2, ..., N_p\}$, and denote each particle as $S^{(p)}$ where $p \in \mathcal{I}$. These particles can be uniquely partitioned according to the partition $\mathcal{P}_i$ introduced for a *setwise conditional* q_c^i in Definition 4.2.1. Specifically, for each $A \in \mathcal{P}_i$, the set of labels of the particles that lie in the region A can be denoted by $F(A) = \{p \in \mathcal{I} : S^{(p)} \in A\}$, and the partition of particle index set $\mathcal{I}$ which is resulted from $\mathcal{P}_i$ can be denoted by $\mathcal{S}_i = \{F(A) : A \in \mathcal{P}_i \wedge F(A) \neq \emptyset\}$. Note that by such a construction, we can easily verify that 1) the $\mathcal{S}_i$ is always a rigorous partition of the index set $\mathcal{I}$, satisfying the definition of set partition in Remark 1 in Section 4.2.1;

and 2) the *conditional everywhere* q_c^i which assumes the *finest* $\mathcal{P}_i$ corresponds to the *finest* index partition $\mathcal{S}_i = \{\{p\} : p \in \mathcal{I}\}$.

Since the optimal global distribution $\tilde{q}_g$ in (4.7) is known only up to a normalisation constant, the self-normalised importance sampling is resorted. Define the unnormalised weight $\tilde{\omega}(S)$ as the ratio between the unnormalised optimal global distribution $\tilde{q}_g(S)$ (i.e. the right hand side of (4.7)) and the proposal $\lambda(S)$, that is

$$\tilde{\omega}(S) = \frac{\exp\left(\mathrm{E}_{\prod_{i=1}^{N_c} q_c^i(Z_i|S)} \log p(S, Z, Y)\right)}{\lambda(S) \prod_{i=1}^{N_c} \exp\left(\mathrm{E}_{q_c^i(Z_i|S)} \log q_c^i(Z_i|S)\right)}. \tag{4.11}$$

Then for each particle $S^{(p)}$, $p \in \mathcal{I}$, its unnormalised weight $\tilde{\omega}(S^{(p)})$, which we abbreviate as $\tilde{\omega}^{(p)}$, can be computed as follows:

$$\tilde{\omega}^{(p)} = \frac{\exp\left(\mathrm{E}_{\prod_{i=1}^{N_c} q_c^i(Z_i|S^{(p)})} \log p(S^{(p)}, Z, Y)\right)}{\lambda(S^{(p)}) \prod_{i=1}^{N_c} \exp\left(\mathrm{E}_{q_c^i(Z_i|S^{(p)})} \log q_c^i(Z_i|S^{(p)})\right)}. \tag{4.12}$$

By normalising the weight $\omega^{*(p)} = \tilde{\omega}^{(p)} / \sum_p \tilde{\omega}^{(p)}$, we can approximate the $\tilde{q}_g(S)$ in (4.7) by an empirical distribution: $\tilde{q}_g(S) \approx \sum_{p=1}^{N_p} \omega^{*(p)} \delta(S^{(p)})$. The update step for q_g in the theoretical CVB (Algorithm 7) can then be approximated by updating the $\tilde{\omega}^{(p)}$ according to (4.12) for all particles $S^{(p)}$.

We now consider the update for $q_c^i(Z_i|S)$. Instead of evaluating $q_c^i(Z_i|S)$ for all possible values of S, we only tackle the associated local distribution $q_c^i(Z_i|S = S^{(p)})$ for each particle $S^{(p)}$ since they are sufficient to calculate the weight in (4.12) to update q_g, and to produce the required posterior approximation, e.g. (4.14). Now for a particular particle $S^{(p)}$, suppose A is the set member of region partition $\mathcal{P}_i$ that includes $S^{(p)}$, i.e. $S^{(p)} \in A \in \mathcal{P}_i$, and B is the corresponding set member of index partition $\mathcal{S}_i$ that includes p , i.e. $p \in B \in \mathcal{S}_i$. Recall that the optimal distribution $\tilde{q}_c^i(Z_i|S = S^{(p)})$ can be any distribution if $q_g(S) = 0$ for all A; otherwise depending on whether A is a singleton set, the optimal $\tilde{q}_c^i(Z_i|S = S^{(p)})$ is either (4.10) or equal to the $\tilde{q}_c^{i,A}(Z_i)$ in (4.9), where the latter is often analytically intractable. Here we approximate the optimal distribution $\tilde{q}_c^i(Z_i|S = S^{(p)})$ by $\hat{q}_c^{i,B}(Z_i)$ where

$$\hat{q}_c^{i,B}(Z_i) \propto \exp\left(\frac{\sum_{j \in B} \tilde{\omega}^{(j)} \mathrm{E}_{q_c^{i-}(Z_{i-}|S^{(j)})} \log p(S^{(j)}, Z, Y)}{\sum_{j \in B} \tilde{\omega}^{(j)}}\right), \tag{4.13}$$

if $\sum_{j \in B} \tilde{\omega}^{(j)} \neq 0$; otherwise $\hat{q}_c^{i,B}(Z_i)$ can be any distribution. Observe that $\hat{q}_c^{i,B}$ in (4.13) is the exact optimal $\tilde{q}_c^i(Z_i|S = S^{(p)})$ in (4.10) if q_c^i is *conditional everywhere* and/or

Algorithm 8: IS-CVB

Require: Particles $S^{(p)} \sim \lambda(S)$ for all $p \in \mathcal{I}$, maximum iteration limit M, tolerance threshold $\epsilon > 0$, initial index partitions $\mathcal{S}_i (i = 1, 2, ..., N_c)$ and initial distributions $\{q\}_{cf}$.
Output: $\tilde{\omega}^{(p)}, q_c^i(Z_i|S^{(p)})$ for all $p \in \mathcal{I}$.
for $k = 1, 2, ..., M$ **do**
  Update $\tilde{\omega}^{(p)}$ according to (4.12) for all $p \in \mathcal{I}$.
  Evaluate the estimated ELBO $\hat{\mathcal{F}}_k$ according to (4.15).
  if $(\hat{\mathcal{F}}_k - \hat{\mathcal{F}}_{k-1}) < \epsilon \wedge k \geq 2$ **then**
    break
  end
  for $i = 1, 2, ..., N_c$ **do**
    Refine the $\mathcal{S}_i$ if higher accuracy is required.
    for $B \in \mathcal{S}_i$ **do**
      if $\sum_{j \in B} \tilde{\omega}^{(j)} = 0$ **then**
        Set $\hat{q}_c^{i,B}$ to an arbitrary distribution.
      else
        Evaluate $\hat{q}_c^{i,B}$ according to (4.13).
      end
      Set $q_c^i(Z_i|S^{(p)}) \leftarrow \hat{q}_c^{i,B}$ for all $p \in B$.
    end
  end
end

$A = \{S^{(p)}\}$ is a singleton set such that $B = \{p\}$; otherwise $\hat{q}_c^{i,B}(Z_i)$ in (4.13) is an approximation of $\tilde{q}_c^{i,A}(Z_i)$ in (4.9), where the exponent of (4.9) is approximated within the importance sampling framework. More details about this approximation and the derivation of (4.13) will be discussed in Section 4.4.2. Additionally, the tractability of this local update (4.13) will be discussed in Section 4.3.3.

Finally, we summarise the importance sampling based CVB (denoted as IS-CVB) in Algorithm 8, and the output can be extracted to produce the required approximated posterior, e.g. the joint distribution $p(Z_1, Z_2)$ can be approximated as the following mixture distributions where the dependence between Z_1 and Z_2 are retained.

$$p(Z_1, Z_2) \approx \sum_{p=1}^{N_p} \omega^{*(p)} q_c^1(Z_1|S^{(p)}) q_c^2(Z_2|S^{(p)}). \tag{4.14}$$

4.3.2 Estimated ELBO

The IS-CVB (Algorithm 8) can offer an estimation of the exact ELBO as a by-product. Specifically, the exact ELBO can be biasedly estimated by $\hat{\mathcal{F}}$, which is defined as follows:

$$\hat{\mathcal{F}} = \log\left(\frac{1}{N_p}\sum_{p=1}^{N_p}\tilde{\omega}^{(p)}\right). \tag{4.15}$$

Theorem 4.3.1. *Suppose $q_c^1, q_c^2, \ldots, q_c^{N_c}$ are the local variational distributions used to compute the unnormalised weight $\tilde{\omega}^{(p)}$ in* (4.12)*, and are also used to compute the exact optimal distribution $\tilde{q}_g(S)$ in* (4.7)*. If we further assume that the samples $S^{(p)}(p \in \mathcal{I})$ drawn from $\lambda(S)$ and employed in* (4.12) *for calculating the $\tilde{\omega}^{(p)}$ are independent of the parametric forms of $q_c^1, q_c^2, \ldots, q_c^{N_c}$, then we have:*

$$\mathrm{E}_{\prod_{p=1}^{N_p}\lambda(S^{(p)})}\exp\left(\hat{\mathcal{F}}\right) = \exp\left(\mathcal{F}\left(\tilde{q}_g, q_c^1, q_c^2, ..., q_c^{N_c}\right)\right),$$

$$\mathrm{E}_{\prod_{p=1}^{N_p}\lambda(S^{(p)})}\hat{\mathcal{F}} \leq \mathcal{F}\left(\tilde{q}_g, q_c^1, q_c^2, ..., q_c^{N_c}\right),$$

where $\hat{F}$ is the estimated ELBO in (4.15)*, and $\mathcal{F}$ is the exact ELBO defined in* (4.2)*.*

The proof is presented in Section 4.4.3. Theorem 4.3.1 suggests that the average of unnormalised weights is an unbiased estimate of the exponential of exact ELBO, and $\hat{\mathcal{F}}$ can be viewed as an unbiased estimate of exact ELBO's lower bound. Similar ELBO estimators have been used as the optimisation objective and performance metric in [139–141], where some derived properties also applies to our case, e.g. when N_p is larger, the $\mathrm{E}_{\prod_p \lambda(S^{(p)})}\hat{\mathcal{F}}$ is tighter to the exact ELBO $\mathcal{F}$. The reader is referred to [139] for details, however note that a major difference in their setting is that they are optimising the ELBO numerically for the variational distribution with predefined parametric form rather than using the *free-form* coordinate ascent update considered here.

In general, the estimated ELBO $\hat{\mathcal{F}}_k$ produced by Algorithm 8 does not satisfy the assumption that samples $S^{(p)}(p \in \mathcal{I})$ are independent of the parametric forms of $q_c^1, q_c^2, \ldots, q_c^{N_c}$. This is because the algorithm uses the same samples $S^{(p)}(p \in \mathcal{I})$ to update $q_c^1, q_c^2, \ldots, q_c^{N_c}$ at every iteration. Such a construction introduces extra bias due to the assumption mismatch when considering $\hat{\mathcal{F}}_k$ as an estimate of the exact ELBO $\mathcal{F}$, but it leads to other useful properties, such as those that will be presented in Theorem 4.3.2 shortly. If one wishes to obtain an estimate that satisfies the assumption in

Theorem 4.3.1, one can use new samples $S^{(p)}$ for computing $\tilde{\omega}^{(p)}$ in (4.12) and $\hat{\mathcal{F}}$ in (4.15).

Note that the exact ELBO $\mathcal{F}$ is now intractable as we cannot compute the KL divergence between a continuous distribution and its particle approximation. Therefore we will instead monitor the $\hat{\mathcal{F}}$ in (4.15) to assess the convergence. It is well-known that the monotonically increasing property of the exact ELBO produced by the standard CAVI can be used to design the algorithm termination condition and to check the implementation of the algorithm. In fact, the $\hat{\mathcal{F}}$ produced by Algorithm 8 has similar monotonically increasing and convergent properties, see Theorem 4.3.2. Therefore, a tolerance threshold of the increment of $\hat{\mathcal{F}}$ is set in Algorithm 8 to determine whether the convergence is reached. Moreover, it is useful to check whether the $\hat{\mathcal{F}}$ produced by Algorithm 8 always satisfies $\hat{\mathcal{F}}_k - \hat{\mathcal{F}}_{k-1} \geq 0$ for all $k \geq 2$; if not, then the implementation of Algorithm 8 is incorrect.

Theorem 4.3.2. *The estimated ELBO in Algorithm 8 is monotonically increasing across iterations, i.e.* $\hat{\mathcal{F}}_k - \hat{\mathcal{F}}_{k-1} \geq 0$ *for all* $k \geq 2$. *Moreover, the sequence* $(\hat{\mathcal{F}}_k)_{k\in\mathbb{N}}$ *is convergent if index partitions* $\mathcal{S}_i (i = 1, 2, ..., N_c)$ *are fixed after some iteration number.*

Remark. Obviously, the convergence of $(\hat{\mathcal{F}}_k)_{k\in\mathbb{N}}$ assumes that Algorithm 8 will not be terminated (e.g. set $M = \infty, \epsilon < 0$) to ensure the existence of the sequence $(\hat{\mathcal{F}}_k)_{k\in\mathbb{N}}$.

The elaborate proof is presented in Section 4.4.4. Similar to the performance improvement achieved by the theoretical CVB (Algorithm 7), the IS-CVB (Algorithm 8) can also guarantee a higher $\hat{\mathcal{F}}$ compared to a more standard Monte Carlo-based mean-field CAVI. The latter can be considered as a special case of Algorithm 8 where $\mathcal{S}_i = \{\mathcal{I}\}$ for all $i = 1, 2, ..., N_c$. When the produced $\hat{\mathcal{F}}$ in such a setting is converged (convergence is guaranteed by Theorem 4.3.2), we can refine the $\mathcal{S}_i$ within the proposed CVB framework. This will increase the $\hat{\mathcal{F}}$ again according to Theorem 4.3.2. The highest $\hat{\mathcal{F}}$ can be achieved by $\mathcal{S}_i$ being the *finest* partition, i.e. the *conditional everywhere* q_c^i. However, each a *conditional everywhere* q_c^i requires computing N_p *free-form* variational distributions in Algorithm 8. When the computational power is limited, the more flexible *setwise conditional* q_c^i is favored as it can achieve a competitive performance with fewer *free-form* variational distributions to evaluate by using a wisely chosen partition $\mathcal{S}_i$.

4.3.3 Tractability of the approximate local update and applicability of IS-CVB

As can be seen in Algorithm 8, the non-trivial computations come only from the weight calculation in (4.12) and the (approximate) local updates in (4.13). In particular, they both require the evaluation of expectation with respect to some local distributions $q_c^i(Z_i|S^{(p)})$; and according to the line 13 in Algorithm 8, $q_c^i(Z_i|S^{(p)})$ equals to $\hat{q}_c^{i,B}$ for the corresponding B that satisfy the $p \in B \in \mathcal{S}_i$. This raises concerns about whether the approximate local updates for $\hat{q}_c^{i,B}$ in (4.13) can yield a distribution with a well-known parametric form for each $B \in \mathcal{S}_i$ and $i = 1, 2, ..., N_c$, since if it cannot, computing expectations with respect to $\hat{q}_c^{i,B}$ would usually be very difficult.

Here we consider a useful and general form for our model $p(X, Y)$ in which the approximate local updates of $\hat{q}_c^{i,B}$ in (4.13) can be easily evaluated in a closed form. Our model assumption for $p(X, Y)$ is closely related to that of *complete conditional in the exponential family* [23], which results in a closed-form expression for the coordinate ascent update in (4.4) for the classical mean-field CAVI. However, our model assumption is less restrictive and can ensure the tractability of the algorithm due to the Monte Carlo approximations for S and the *setwise conditional* assumptions for q_c^i, as will be shown later.

Suppose that for all $S \in A$ where $A \in \mathcal{P}_i$, the conditional $p(Z_i|Z_{i-}, S, Y)$ can be expressed in the exponential family, i.e.

$$p(Z_i|Z_{i-}, S, Y) \propto h_{i,A}(Z_i) \exp\left(\eta_{i,A}(Z_{i-}, S, Y)^\top T_{i,A}(Z_i)\right) \qquad \textit{for all } \; S \in A, \quad (4.16)$$

where $h_{i,A}$ is the base measure, $T_{i,A}$ is the sufficient statistics, and the natural parameter $\eta_{i,A}$ is a function of the conditioning set $\{Z_{i-}, S, Y\}$. Then the parametric form of $\hat{q}_c^{i,B}(Z_i)$ in our local update (4.13) for the corresponding $B = \{p \in \mathcal{I} : S^{(p)} \in A\}$ (here $B \neq \emptyset$ is obviously assumed) is the same member in the exponential family. Specifically,

it can be derived as follows

$$\begin{aligned}
&\hat{q}_c^{i,B}(Z_i) \propto \exp\left(\frac{\sum_{j\in B}\tilde{\omega}^{(j)}\mathrm{E}_{q_c^{i-}(Z_{i-}|S^{(j)})}\log p(S^{(j)},Z,Y)}{\sum_{j\in B}\tilde{\omega}^{(j)}}\right)\\
=&\exp\left(\frac{\sum_{j\in B}\tilde{\omega}^{(j)}\mathrm{E}_{q_c^{i-}(Z_{i-}|S^{(j)})}\left[\log p(Z_i|Z_{i-},S^{(j)},Y)+\log p(Z_{i-},S^{(j)},Y)\right]}{\sum_{j\in B}\tilde{\omega}^{(j)}}\right)\\
=&\exp\left(\frac{\sum_{j\in B}\tilde{\omega}^{(j)}\mathrm{E}_{q_c^{i-}(Z_{i-}|S^{(j)})}\left[\log h_{i,A}(Z_i)+\eta_{i,A}(Z_{i-},S^{(j)},Y)^\top T_{i,A}(Z_i)+c(Z_{i-},S^{(j)},Y)\right]}{\sum_{j\in B}\tilde{\omega}^{(j)}}\right)\\
\propto&\exp\left(\log h_{i,A}(Z_i)+\frac{\sum_{j\in B}\tilde{\omega}^{(j)}\left[\mathrm{E}_{q_c^{i-}(Z_{i-}|S^{(j)})}\eta_{i,A}(Z_{i-},S^{(j)},Y)\right]^\top}{\sum_{j\in B}\tilde{\omega}^{(j)}}T_{i,A}(Z_i)\right)\\
=&h_{i,A}(Z_i)\exp\left(\frac{\sum_{j\in B}\tilde{\omega}^{(j)}\left[\mathrm{E}_{q_c^{i-}(Z_{i-}|S^{(j)})}\eta_{i,A}(Z_{i-},S^{(j)},Y)\right]^\top}{\sum_{j\in B}\tilde{\omega}^{(j)}}T_{i,A}(Z_i)\right),
\end{aligned}\tag{4.17}$$

where $c(Z_{i-},S^{(j)},Y)$ is a function of $Z_{i-},S^{(j)},Y$ and is absorbed as a normalisation constant in the next line. Due to the fact that for all $j\in B$, we have $S^{(j)}\in A$, the third line in (4.17) can thus be deduced by substituting $p(Z_i|Z_{i-},S^{(j)},Y)$ with (4.16). We can see that $\hat{q}_c^{i,B}(Z_i)$ admits the same base measure $h_{i,A}(Z_i)$ and sufficient statistics $T_{i,A}(Z_i)$ as in the original complete conditional $p(Z_i|Z_{i-},S,Y)$, whilst the new natural parameter is a weighted summation of $\mathrm{E}_{q_c^{i-}(Z_{i-}|S^{(j)})}\eta_{i,A}(Z_{i-},S^{(j)},Y)$, i.e. the expectations of the original natural parameters. One might anticipate that all of these expectations can be evaluated analytically once all other local distributions q_c^{i-} can be updated similarly in the exponential family.

To ensure that the local updates $\hat{q}_c^{i,B}(Z_i)$ in (4.13) can be easily evaluated in the exponential family for all $i=1,2,...,N_c$ and all $B\in\mathcal{S}_i$ no matter what particles $S^{(p)}$ $(p=1,2,...,N_p)$ are drawn, then the assumption (4.16) should apply to all local variables' conditional $p(Z_i|Z_{i-},S,Y)$ $(i=1,2,...,N_c)$ when $S\in A$ for each $A\in\mathcal{P}_i$. This requires that the local variables' conditional $p(Z_i|Z_{i-},S,Y)$ in our model can be expressed as follows for each $i=1,2,...,N_c$:

$$p(Z_i|Z_{i-},S,Y)=\sum_{A\in\mathcal{P}_i}I(S\in A)\frac{h_{i,A}(Z_i)\exp\left(\eta_{i,A}(Z_{i-},S,Y)^\top T_{i,A}(Z_i)\right)}{\int h_{i,A}(Z_i)\exp\left(\eta_{i,A}(Z_{i-},S,Y)^\top T_{i,A}(Z_i)\right)dZ_i},\tag{4.18}$$

where the integral in the denominator is the normalisation constant that does not depend on Z_i and ensures the conditional density integrals to 1. We can see that such a conditional density reduces to the standard exponential family of argument Z_i if $h_{i,A},\eta_{i,A},T_{i,A}$ are equal over different $A\in\mathcal{P}_i$. This form enables tractable updates in

the classical mean-field CAVI. Our IS-CVB demonstrates increased applicability as the model assumption in (4.18) is more general, and this constraint applies only to local variables Z_i, not the global variable S.

Our model assumption in (4.18) also suggests that the refining $\mathcal{P}_i$ in Algorithm 8, i.e. adopting a *finer* partition $\mathcal{P}_i$, will never cause a potential intractability for updating Z_i. In the contrast, a *finer* partition $\mathcal{P}_i$ can only enhance the tractability of local update. This is because a finer $\mathcal{P}_i$ renders the model assumption in (4.18) more general, encompassing all models that satisfy the previous assumption induced by a coarser $\mathcal{P}_i$. Such a phenomenon suggests the following approach to wisely select the global variable S, local variable Z_i, and partition $\mathcal{P}_i$ $(i = 1, 2, ..., N_c)$, from the perspective of ensuring the tractability of the IS-CVB: initially, with a global variable S, one can test various factorised local variables Z_i to verify if the model assumption in (4.18) is satisfied. If not, a finer partition $\mathcal{P}_i$ can be attempted for the local variable Z_i that does not meet (4.18). If (4.18) remains unmet, it may be necessary to incorporate variable Z_i into the global variable S. This will bypass the restriction in (4.18) for this variable, but the dimensions of sampled variable will accordingly increase.

4.4 Detailed derivations

This section presents detailed proofs for three theorems introduced in Section 4.2.2 and 4.3.2, and the derivation for the approximate local update (4.13). First we present the following lemma that will be frequently used in this thesis.

Lemma 4.4.1. *Let $X \in \mathbb{R}^d$ be a random variable, and $q(X)$ be a probability distribution of it. Suppose $f : \mathbb{R}^d \to \mathbb{R}$ is a function of X such that $\int_{\mathbb{R}^d} \exp(f(X))\, dX < \infty$, then we have*

$$\mathrm{E}_{q(X)}\left[-\log q(X) + f(X)\right] = -\mathrm{KL}(q(X)||\check{f}(X)) + c, \tag{4.19}$$

where $\check{f}(X)$ is another density function of X, and c is a constant that does not depend on X. Both are defined below:

$$\check{f}(X) = \frac{\exp(f(X))}{\int_{\mathbb{R}^d} \exp(f(X))\, dX}, \tag{4.20}$$

$$c = \log \int_{\mathbb{R}^d} \exp(f(X))\, dX. \tag{4.21}$$

Remark 1. The random variable X can be discrete, continuous or mixed type, and the distribution $q(X)$ should be interpreted as PMF and generalised PDF accordingly. The integral in (4.20) and (4.21) should be replaced by summation if X is discrete variable.

Remark 2. The sufficient condition $\int_{\mathbb{R}^d} \exp(f(X))\, dX < \infty$ is placed in Lemma in order to ensure the distribution $\check{f}(X)$ constructed in (4.20) is always valid. This condition rules out cases where $\exp(f(X))$ cannot be normalised to be a valid density, such as when $f(X) = X$.

Proof. By direct rewriting $\mathrm{E}_{q(X)}[-\log q(X) + f(X)]$, we have

$$\begin{aligned}
&\mathrm{E}_{q(X)}[-\log q(X) + f(X)] = \mathrm{E}_{q(X)}[-\log q(X) + \log \exp(f(X))] \\
=&\mathrm{E}_{q(X)}\left[-\log q(X) + \log \frac{\exp(f(X))}{\int_{\mathbb{R}^d} \exp(f(X))\, dX} + \log \int_{\mathbb{R}^d} \exp(f(X))\, dX\right]. \qquad (4.22)
\end{aligned}$$

Since $\exp(f(X))$ is non-negative, and by definition, the integral $\int_{\mathbb{R}^d} \exp(f(X))\, dX$ is finite, we can then construct a density function by normalising $\exp(f(X))$ to satisfy the condition that it integrates to 1. Such a construction leads to the density function $\check{f}(X)$ defined in (4.20). Subsequently, (4.22) can be expressed as

$$\begin{aligned}
&\mathrm{E}_{q(X)}[-\log q(X) + f(X)] = \mathrm{E}_{q(X)}\left[-\log q(X) + \log \check{f}(X) + \log \int_{\mathbb{R}^d} \exp(f(X))\, dX\right] \\
=&\mathrm{E}_{q(X)}\left[-\log q(X) + \log \check{f}(X)\right] + \log \int_{\mathbb{R}^d} \exp(f(X))\, dX \\
=&-\mathrm{KL}(q(X)||\check{f}(X)) + c, \qquad (4.23)
\end{aligned}$$

where the last line is obtained by using the definition of KL divergence and c defined in (4.21).

□

This Lemma 4.4.1 will be frequently employed in our derivation, and we will always check whether the requirement $\int_{\mathbb{R}^d} \exp(f(X))\, dX < \infty$ is satisfied in order to ensure the distribution $\check{f}(X)$ in (4.20) always exists and Lemma can be employed safely.

4.4.1 Proof of Theorem 4.2.1

Here we present the proof for Theorem 4.2.1 on page 105. In this section, E_{q_g} serves as a shorthand for $E_{q_g(S)}$ to make the formula more compact. Denote the optimal global distribution by $\tilde{q}_g(S) = \arg\max_{q_g} \mathcal{F}(\{q\}_{cf})$, and the optimal $q_c^i(Z_i|S)$ by $\tilde{q}_c^i(Z_i|S) = \arg\max_{q_c^i} \mathcal{F}(\{q\}_{cf})$ for $i = 1, 2, ..., N_c$. In adherence to the convention

established in Theorem 4.2.1, the integral employed within this section is similarly generalised: if the associated variable is discrete, it should be substituted with a summation.

4.4.1.1 Derivation of the optimiser $\tilde{q}_g$

We first prove (4.7). Rewrite the ELBO $\mathcal{F}(\{q\}_{cf})$ by substituting (4.5) into (4.2),

$$\begin{aligned}\mathcal{F}(\{q\}_{cf}) =& \mathrm{E}_{q_g}\mathrm{E}_{\prod_{i=1}^{N_c} q_c^i(Z_i|S)} \log \frac{p(X,Y)}{\prod_{i=1}^{N_c} q_c^i(Z_i|S)} - \mathrm{E}_{q_g} \log q_g(S) \\ =& \mathrm{E}_{q_g}\left[\mathrm{E}_{\prod_{i=1}^{N_c} q_c^i(Z_i|S)} \log \frac{p(X|Y)}{\prod_{i=1}^{N_c} q_c^i(Z_i|S)} - \log q_g(S)\right] + \log p(Y). \end{aligned} \tag{4.24}$$

In order to apply Lemma 4.4.1, we first need to verify that the following integral is finite:

$$\int_U \exp\left(\mathrm{E}_{\prod_{i=1}^{N_c} q_c^i(Z_i|S)} \log \frac{p(X|Y)}{\prod_{i=1}^{N_c} q_c^i(Z_i|S)}\right) dS. \tag{4.25}$$

Note that by Jensen's inequality, we have

$$\begin{aligned}\mathrm{E}_{\prod_{i=1}^{N_c} q_c^i(Z_i|S)} \log \frac{p(X|Y)}{\prod_{i=1}^{N_c} q_c^i(Z_i|S)} &\leq \log \mathrm{E}_{\prod_{i=1}^{N_c} q_c^i(Z_i|S)} \frac{p(X|Y)}{\prod_{i=1}^{N_c} q_c^i(Z_i|S)} \\ &= \int p(Z_1, Z_2, ..., Z_{N_c}, S|Y) dZ_1 dZ_2 ... dZ_{N_c} \\ &= \log p(S|Y). \end{aligned} \tag{4.26}$$

Then, since the integrand in (4.25) is non-negative, and $\exp(\cdot)$ is monotonically increasing, we have

$$\int_U \exp\left(\mathrm{E}_{\prod_{i=1}^{N_c} q_c^i(Z_i|S)} \log \frac{p(X|Y)}{\prod_{i=1}^{N_c} q_c^i(Z_i|S)}\right) dS \leq \int_U \exp\left(\log p(S|Y)\right) dS = 1, \tag{4.27}$$

which is a finite number. We can then conclude that the integral in (4.25) is finite, and safely apply Lemma 4.4.1 to express the ELBO in (4.24) as

$$\begin{aligned}\mathcal{F}(\{q\}_{cf}) =& \mathrm{E}_{q_g}\left[\mathrm{E}_{\prod_{i=1}^{N_c} q_c^i(Z_i|S)} \log \frac{p(X|Y)}{\prod_{i=1}^{N_c} q_c^i(Z_i|S)} - \log q_g(S)\right] + \log p(Y) \\ =& -\mathrm{KL}(q_g(S)||\tilde{p}(S)) + c + \log p(Y), \end{aligned} \tag{4.28}$$

$$\begin{aligned}\tilde{p}(S) =& \exp\left(\mathrm{E}_{\prod_{i=1}^{N_c} q_c^i(Z_i|S)} \log \frac{p(X|Y)}{\prod_{i=1}^{N_c} q_c^i(Z_i|S)}\right) \frac{1}{\exp(c)} \\ =& \frac{\exp\left(\mathrm{E}_{\prod_{i=1}^{N_c} q_c^i(Z_i|S)} \log p(X|Y)\right)}{\prod_{i=1}^{N_c} \exp\left(\mathrm{E}_{q_c^i(Z_i|S)} \log q_c^i(Z_i|S)\right)} \frac{1}{\exp(c)},\end{aligned} \tag{4.29}$$

$$c = \log \int_U \exp\left(\mathrm{E}_{\prod_{i=1}^{N_c} q_c^i(Z_i|S)} \log \frac{p(X|Y)}{\prod_{i=1}^{N_c} q_c^i(Z_i|S)}\right) dS \leq 0. \tag{4.30}$$

where c is a log normalisation factor which does not depend on S. Moreover, when all q_c^i are fixed, c is a constant. Using (4.28) and the fact that $p(Y)$ is a constant that does not depend on q_g, we have

$$\tilde{q}_g(S) = \arg\max_{q_g} \mathcal{F}(\{q\}_{cf}) = \arg\max_{q_g} \left(-\mathrm{KL}(q_g(S)||\tilde{p}(S))\right).$$

With the definition that q_g is *free-form*, q_g can take any parametric form. Hence $q_g(S) = \tilde{p}(S)$, which minimises the KL divergence above, is the unique solution for the optimal $\tilde{p}(S)$, i.e.

$$\tilde{q}_g(S) = \tilde{p}(S) \propto \frac{\exp\left(\mathrm{E}_{\prod_{i=1}^{N_c} q_c^i(Z_i|S)} \log p(X|Y)\right)}{\prod_{i=1}^{N_c} \exp\left(\mathrm{E}_{q_c^i(Z_i|S)} \log q_c^i(Z_i|S)\right)}. \tag{4.31}$$

This optimiser in (4.31), by multiplying a constant $p(Y)$, concurs with (4.7) in Theorem 4.2.1. Thus, we have successfully derived the unique optimiser $\tilde{q}_g(S)$ in Theorem 4.2.1.

We now start to derive the optimiser $\tilde{q}_c^i(Z_i|S)$ in Theorem 4.2.1. First, we rewrite the ELBO $\mathcal{F}(\{q\}_{cf})$ in equation (4.24) as:

$$\begin{aligned}\mathcal{F}(\{q\}_{cf}) =& \mathrm{E}_{q_g} \mathrm{E}_{q_c^i(Z_i|S)} \left(\mathrm{E}_{q_c^{i-}(Z_{i-}|S)} \log p(Z_i|Z_{i-}, S, Y) - \log q_c^i(Z_i|S)\right) \\ &+ \mathrm{E}_{q_g} \mathrm{E}_{q_c^{i-}(Z_{i-}|S)} p(Z_{i-}, S, Y) - \mathrm{E}_{q_g} \log q_g(S) - \mathrm{E}_{q_g} \mathrm{E}_{q_c^{i-}(Z_{i-}|S)} \log q_c^{i-}(Z_{i-}|S),\end{aligned} \tag{4.32}$$

where

$$q_c^{i-}(Z_{i-}|S) = \begin{cases} \prod_{l \neq i} q_c^l(Z_l|S) & N_c > 1 \\ 1 & N_c = 1 \end{cases}$$

$$E_{q_c^{i-}(Z_{i-}|S)}[\cdot] = \begin{cases} \int [\cdot] \prod_{l \neq i} q_c^l(Z_l|S) dZ_{i-} & N_c > 1 \\ [\cdot] & N_c = 1 \end{cases}$$

Notice the last three terms in (4.32) are constants when q_g and q_c^{i-} are fixed, hence we have

$$\tilde{q}_c^i(Z_i|S) = \arg\max_{q_c^i} \mathcal{F}(\{q\}_{cf}) = \arg\max_{q_c^i} \mathrm{E}_{q_g} F(S, q_c), \tag{4.33}$$

where we define

$$F(S, q_c) = \mathrm{E}_{q_c^i(Z_i|S)} \left(\mathrm{E}_{q_c^{i-}(Z_{i-}|S)} \log p(Z_i|Z_{i-}, S, Y) - \log q_c^i(Z_i|S) \right).$$

4.4.1.2 Derivation of the *setwise conditional* optimiser $\tilde{q}_c^i$

We now derive the *setwise conditional* $\tilde{q}_c^i$ and prove (4.9). Note that $\mathrm{E}_{q_g} F(S, q_c)$ in (4.33) can be expressed as follows by using (4.6) in the Definition 4.2.1 on page 104,

$$\begin{aligned}
&\mathrm{E}_{q_g} F(S, q_c) \\
=&\mathrm{E}_{q_g} \int \sum_{A \in \mathcal{P}_i} q_c^{i,A}(Z_i) I(S \in A) \left[\mathrm{E}_{q_c^{i-}(Z_{i-}|S)} \log p(Z_i|Z_{i-}, S, Y) - \log \sum_{B \in \mathcal{P}_i} q_c^{i,B}(Z_i) I(S \in B) \right] dZ_i \\
=&\mathrm{E}_{q_g} \int \sum_{A \in \mathcal{P}_i} q_c^{i,A}(Z_i) I(S \in A) \left[\mathrm{E}_{q_c^{i-}(Z_{i-}|S)} \log p(Z_i|Z_{i-}, S, Y) - \sum_{B \in \mathcal{P}_i} I(S \in B) \log q_c^{i,B}(Z_i) \right] dZ_i \\
=&\mathrm{E}_{q_g} \int \sum_{A \in \mathcal{P}_i} q_c^{i,A}(Z_i) I(S \in A) \left[\mathrm{E}_{q_c^{i-}(Z_{i-}|S)} \log p(Z_i|Z_{i-}, S, Y) - \log q_c^{i,A}(Z_i) \right] dZ_i \\
=&\mathrm{E}_{q_g} \sum_{A \in \mathcal{P}_i} \mathrm{E}_{q_c^{i,A}(Z_i)} I(S \in A) \left[\mathrm{E}_{q_c^{i-}(Z_{i-}|S)} \log p(Z_i|Z_{i-}, S, Y) - \log q_c^{i,A}(Z_i) \right] \\
=& \sum_{A \in \mathcal{P}_i} \mathrm{E}_{q_c^{i,A}(Z_i)} \int_U q_g(S) I(S \in A) \left[\mathrm{E}_{q_c^{i-}(Z_{i-}|S)} \log p(Z_i|Z_{i-}, S, Y) - \log q_c^{i,A}(Z_i) \right] dS \\
=& \sum_{A \in \mathcal{P}_i} \mathrm{E}_{q_c^{i,A}(Z_i)} \int_A q_g(S) \left[\mathrm{E}_{q_c^{i-}(Z_{i-}|S)} \log p(Z_i|Z_{i-}, S, Y) - \log q_c^{i,A}(Z_i) \right] dS \\
=& \sum_{A \in \mathcal{P}_i} \mathrm{E}_{q_c^{i,A}(Z_i)} \left[\int_A q_g(S) \mathrm{E}_{q_c^{i-}(Z_{i-}|S)} \log p(Z_i|Z_{i-}, S, Y) dS - \left(\int_A q_g(S) dS \right) \log q_c^{i,A}(Z_i) \right],
\end{aligned} \tag{4.34}$$

where the third equality is obtained by noting that each element in the partition $\mathcal{P}_i$ is disjoint with each other, i.e. $I(S \in A) I(S \in B)$ is 1 when $A = B$, and zero otherwise. The second equality is obtained from the fact that $\log \sum_{B \in \mathcal{P}_i} q_c^{i,B}(Z_i) I(S \in B)$ equals to $\sum_{B \in \mathcal{P}_i} I(S \in B) \log q_c^{i,B}(Z_i)$. To see this, first recall that S has support U, and $\mathcal{P}_i$ is a partition of U. Let's represent the elements of $\mathcal{P}_i$ as $V_1, V_2, ..., V_N$ (i.e. $\mathcal{P}_i = \{V_1, V_2, ..., V_N\}$ where N may be infinite), then $\mathcal{P}_i$ is a partition of U implies (from Remark 1 of Definition 4.2.1) that $\bigcup_{j=1,2,...,N} V_j = \mathcal{P}_i$, and $V_j \bigcap V_k = \emptyset$ if $j \neq k$.

This ensures that no matter what value S takes, S will belong to only one of the V_j sets and to no other element in $\mathcal{P}_i$ at the same time. Then we can find both of $\log \sum_{B\in\mathcal{P}_i} q_c^{i,B}(Z_i) I(S \in B)$ and $\sum_{B\in\mathcal{P}_i} I(S \in B) \log q_c^{i,B}(Z_i)$ can be exactly rewritten as the same step function in (4.35) that depends on the values of S, i.e.

$$\begin{aligned}
\log \sum_{B\in\mathcal{P}_i} q_c^{i,B}(Z_i) I(S \in B) &= \log \sum_{B\in\{V_1,V_2,\ldots,V_N\}} q_c^{i,B}(Z_i) I(S \in B) \\
&= \begin{cases} \log q_c^{i,V_1}(Z_i), & \text{if } S \in V_1; \\ \log q_c^{i,V_2}(Z_i), & \text{if } S \in V_2; \\ \ldots & \ldots \\ \log q_c^{i,V_N}(Z_i), & \text{if } S \in V_N. \end{cases} \\
&= \sum_{B\in\{V_1,V_2,\ldots,V_N\}} I(S \in B) \log q_c^{i,B}(Z_i) \\
&= \sum_{B\in\mathcal{P}_i} I(S \in B) \log q_c^{i,B}(Z_i).
\end{aligned} \tag{4.35}$$

Therefore, $\log \sum_{B\in\mathcal{P}_i} q_c^{i,B}(Z_i) I(S \in B)$ equals to $\sum_{B\in\mathcal{P}_i} I(S \in B) \log q_c^{i,B}(Z_i)$, and the second equality of (4.34) is proved.

We can see from (4.34) that each $q_c^{i,A}$ only appears in one summand, and each summand only involves one $q_c^{i,A}$. Furthermore, if there exists some $A \in \mathcal{P}_i$ such that $q_g(S) = 0$ for all $S \in A$, then clearly the summands that correspond to those A will always be zero regardless of the parametric form of $q_c^{i,A}$. This is because for these summands, both terms in the bracket in (4.34) will always be zero. This motives us to continue writing (4.34) as

$$\begin{aligned}
&\mathrm{E}_{q_g} F(S, q_c) \\
&= \sum_{A\in\mathcal{P}_i} \mathrm{E}_{q_c^{i,A}(Z_i)} \left[\int_A q_g(S) \mathrm{E}_{q_c^{i-}(Z_{i-}|S)} \log p(Z_i|Z_{i-}, S, Y) dS - \left(\int_A q_g(S) dS \right) \log q_c^{i,A}(Z_i) \right] \\
&= \sum_{A\in\mathcal{P}_i^g} \mathrm{E}_{q_c^{i,A}(Z_i)} \left[\int_A q_g(S) \mathrm{E}_{q_c^{i-}(Z_{i-}|S)} \log p(Z_i|Z_{i-}, S, Y) dS - \left(\int_A q_g(S) dS \right) \log q_c^{i,A}(Z_i) \right] \\
&= \sum_{A\in\mathcal{P}_i^g} \left(\int_A q_g(S) dS \right) \mathrm{E}_{q_c^{i,A}(Z_i)} \left[\frac{\int_A q_g(S) \mathrm{E}_{q_c^{i-}(Z_{i-}|S)} \log p(Z_i|Z_{i-}, S, Y) dS}{\int_A q_g(S) dS} - \log q_c^{i,A}(Z_i) \right]
\end{aligned} \tag{4.36}$$

where $\mathcal{P}_i^g$ is the set that only includes the partition set A such that $q_g(S)$ is not always zero for all $S \in A$, i.e.

$$\mathcal{P}_i^g = \{A \in \mathcal{P}_i : (\exists S \in A)[q_g(S) \neq 0]\}, \tag{4.37}$$

The first term in the bracket of (4.36) can be finite even if the denominator $\int_A q_g(S)dS$ is infinitesimal. This is because we can always rewrite it as

$$\begin{aligned}
&\frac{\int_A q_g(S)\mathrm{E}_{q_c^{i-}(Z_{i-}|S)} \log p(Z_i|Z_{i-}, S, Y)dS}{\int_A q_g(S)dS} \\
&= \int_U \frac{q_g(S)I(S \in A)}{\int_A q_g(S)dS}\mathrm{E}_{q_c^{i-}(Z_{i-}|S)} \log p(Z_i|Z_{i-}, S, Y)dS \\
&= \int_U q_g^A(S)\mathrm{E}_{q_c^{i-}(Z_{i-}|S)} \log p(Z_i|Z_{i-}, S, Y)dS = \mathrm{E}_{q_g^A q_c^{i-}(Z_{i-}|S)} \log p(Z_i|Z_{i-}, S, Y),
\end{aligned} \tag{4.38}$$

where $q_g^A(S)$ is a 'normalised' global density defined below,

$$q_g^A(S) = \frac{q_g(S)}{\int_A q_g(S)dS} I(S \in A). \tag{4.39}$$

We can check $q_g^A(S)$ is a valid density over the domain U by the assumption that $q_g(S)$ is not always 0 for all $S \in A$. Specifically, $q_g^A(S)$ is non-negative and integrates to 1. Now, in order to use Lemma 4.4.1 to optimise the $\mathrm{E}_{q_g} F(S, q_c)$ in (4.36), we need to check the following integral is finite

$$\int \exp\left(\frac{\int_A q_g(S)\mathrm{E}_{q_c^{i-}(Z_{i-}|S)} \log p(Z_i|Z_{i-}, S, Y)dS}{\int_A q_g(S)dS}\right) dZ_i. \tag{4.40}$$

By using (4.38), we have

$$
\begin{aligned}
&\int \exp\left(\frac{\int_A q_g(S)\mathrm{E}_{q_c^{i-}(Z_{i-}|S)}\log p(Z_i|Z_{i-},S,Y)dS}{\int_A q_g(S)dS}\right)dZ_i \\
=&\int \exp\left(\mathrm{E}_{q_g^A q_c^{i-}(Z_{i-}|S)}\log p(Z_i|Z_{i-},S,Y)\right)dZ_i. \\
=&\int \exp\left(\mathrm{E}_{q_g^A q_c^{i-}(Z_{i-}|S)}\log\frac{p(Z_i,Z_{i-},S|Y)}{q_g^A(S)q_c^{i-}(Z_{i-}|S)}\frac{q_g^A(S)q_c^{i-}(Z_{i-}|S)}{p(Z_{i-},S|Y)}\right)dZ_i \\
=&\int \exp\left(\mathrm{E}_{q_g^A q_c^{i-}(Z_{i-}|S)}\log\frac{q_g^A(S)q_c^{i-}(Z_{i-}|S)}{p(Z_{i-},S|Y)}+\mathrm{E}_{q_g^A q_c^{i-}(Z_{i-}|S)}\log\frac{p(Z_i,Z_{i-},S|Y)}{q_g^A(S)q_c^{i-}(Z_{i-}|S)}\right)dZ_i \\
=&\exp\left(\mathrm{KL}\left(q_g^A(S)q_c^{i-}(Z_{i-}|S)||p(Z_{i-},S|Y)\right)\right)\int \exp\left(\mathrm{E}_{q_g^A q_c^{i-}(Z_{i-}|S)}\log\frac{p(Z_i,Z_{i-},S|Y)}{q_g^A(S)q_c^{i-}(Z_{i-}|S)}\right)dZ_i \\
\leq&\exp\left(\mathrm{KL}\left(q_g^A(S)q_c^{i-}(Z_{i-}|S)||p(Z_{i-},S|Y)\right)\right)\int \exp\left(\log\mathrm{E}_{q_g^A q_c^{i-}(Z_{i-}|S)}\frac{p(Z_i,Z_{i-},S|Y)}{q_g^A(S)q_c^{i-}(Z_{i-}|S)}\right)dZ_i \\
=&\exp\left(\mathrm{KL}\left(q_g^A(S)q_c^{i-}(Z_{i-}|S)||p(Z_{i-},S|Y)\right)\right)\int\int\int p(Z_i,Z_{i-},S|Y)dSdZ_{i-}dZ_i \\
=&\exp\left(\mathrm{KL}\left(q_g^A(S)q_c^{i-}(Z_{i-}|S)||p(Z_{i-},S|Y)\right)\right), \qquad (4.41)
\end{aligned}
$$

where the inequality is obtained similarly as in (4.27), i.e. by using Jensen's inequity, the fact that integrand is non-negative, and the $\exp(\cdot)$ is monotonically increasing. Equation (4.41) suggests that, the integral in (4.40) is bounded above by the exponential of the KL divergence between variational distribution $q_g^A(S)q_c^{i-}(Z_{i-}|S)$ and exact posterior $p(Z_{i-},S|Y)$. By following the standard routine of coordinate ascent update, the $q_g^A(S)q_c^{i-}(Z_{i-}|S)$ will put zero probability mass in regions where the exact posterior $p(Z_{i-},S|Y)$ has zero probability mass, and subsequently, the $\mathrm{KL}\left(q_g^A(S)q_c^{i-}(Z_{i-}|S)||p(Z_{i-},S|Y)\right)$ is typically finite. Therefore, we can safely apply Lemma 4.4.1 to rewrite the objective function in (4.36) as

$$
\begin{aligned}
&\mathrm{E}_{q_g}F(S,q_c) \\
=&\sum_{A\in\mathcal{P}_i^g}\left(\int_A q_g(S)dS\right)\mathrm{E}_{q_c^{i,A}(Z_i)}\left[\frac{\int_A q_g(S)\mathrm{E}_{q_c^{i-}(Z_{i-}|S)}\log p(Z_i|Z_{i-},S,Y)dS}{\int_A q_g(S)dS}-\log q_c^{i,A}(Z_i)\right] \\
=&\sum_{A\in\mathcal{P}_i^g}\left(\int_A q_g(S)dS\right)\left[-\mathrm{KL}\left(q_c^{i,A}(Z_i)||\tilde{p}_A(Z_i)\right)+c_A\right], \qquad (4.42)
\end{aligned}
$$

where

$$\tilde{p}_A(Z_i) \propto \exp\left(\frac{\int_A q_g(S)\mathrm{E}_{q_c^{i-}(Z_{i-}|S)} \log p(Z_i|Z_{i-},S,Y)dS}{\int_A q_g(S)dS}\right), \tag{4.43}$$

$$\begin{aligned} c_A &= \log \int \exp\left(\frac{\int_A q_g(S)\mathrm{E}_{q_c^{i-}(Z_{i-}|S)} \log p(Z_i|Z_{i-},S,Y)dS}{\int_A q_g(S)dS}\right) dZ_i \\ &\leq \mathrm{KL}\left(q_g^A(S)q_c^{i-}(Z_{i-}|S)||p(Z_{i-},S|Y)\right), \end{aligned} \tag{4.44}$$

Now, we are able to consider the optimisation of $\mathrm{E}_{q_g}F(S,q_c)$ in (4.42), which is required in the optimal $\tilde{q}_c^i(Z_i|S)$ in (4.33). Recall that by the definition of *setwise conditional* $q_c^{i,A}$, the optimal $\tilde{q}_c^i(Z_i|S)$ in (4.33) can always be expressed as

$$\tilde{q}_c^i(Z_i|S) = \arg\max_{q_c^i} \mathcal{F}(\{q\}_{cf}) = \arg\max_{q_c^i} \mathrm{E}_{q_g}F(S,q_c) = \sum_{A\in\mathcal{P}_i} \tilde{q}_c^{i,A}(Z_i)I(S\in A), \tag{4.45}$$

where each $\tilde{q}^{i,A}$ can be chosen freely with no relationship with each other. This expression of $\tilde{q}_c^i(Z_i|S)$ in (4.45) matches (4.8) in Theorem 4.2.1. Now, since the value of $\mathrm{E}_{q_g}F(S,q_c)$ in (4.42) does not depend on the $q_c^{i,A}$ when $A \notin \mathcal{P}_i^g$, the optimal distribution $\tilde{q}_c^{i,A}$ for all $A \in \mathcal{P}_i \setminus \mathcal{P}_i^g$ can take any form, since its specific parametric form has no impact on the value of ELBO. Therefore, by using the definition in (4.37), we have the conclusion in Theorem 4.2.1, i.e. $\tilde{q}_c^{i,A}(Z_i)$ can be any distribution if the density $q_g(S) = 0$ for all $S \in A$.

For other A such that $A \in \mathcal{P}_i^g$, the specific form of $q_c^{i,A}$ may indeed affect the value of $\mathrm{E}_{q_g}F(S,q_c)$ in (4.42), and hence a careful optimisation is needed. First note that each summand in (4.42) only involves a unique $q_c^{i,A}$, then the optimisation for each summand can be carried out independently. Furthermore, each summand is a non-negative constant $\int_A q_g(S)dS$ times a function of $q_c^{i,A}$ in the bracket of (4.42). Therefore, when each bracket achieves the maximum, each summand in (4.42) also reaches the maximum, and their sum $\mathrm{E}_{q_g}F(S,q_c)$ must also be maximised. Hence a solution (not necessarily a unique solution) to the optimisation in (4.45) can be obtained by independently optimising each bracket in (4.42). This leads to each $\tilde{q}_c^{i,A}(Z_i)$ in (4.45) being

$$\begin{aligned} \tilde{q}_c^{i,A}(Z_i) =& \arg\max_{q_c^{i,A}} \left[-\mathrm{KL}\left(q_c^{i,A}(Z_i)||\tilde{p}_A(Z_i)\right) + c_A\right] \\ =& \tilde{p}_A(Z_i) \propto \exp\left(\frac{\int_A q_g(S)\mathrm{E}_{q_c^{i-}} \log p(Z_i|Z_{i-},S,Y)dS}{\int_A q_g(S)dS}\right), \end{aligned} \tag{4.46}$$

where we substitute (4.43) to obtain the final result. The second equality in (4.46) is obtained because the parametric form of $q_c^{i,A}(Z_i)$ can be freely chosen by definition, and hence it has to equals to $\tilde{p}_A(Z_i)$ to minimise the KL divergence. At last, by noting that the final result in (4.46) equals the right hand side of (4.9) on page 106 up to a multiplicative constant, i.e.

$$\begin{aligned}
&\tilde{q}_c^{i,A}(Z_i) \propto \exp\left(\frac{\int_A q_g(S)\mathrm{E}_{q_c^{i-}(Z_{i-}|S)} \log p(Z_i|Z_{i-},S,Y)dS}{\int_A q_g(S)dS}\right) \\
=&\exp\left(\frac{\int_A q_g(S)\mathrm{E}_{q_c^{i-}(Z_{i-}|S)} \log p(X,Y)dS}{\int_A q_g(S)dS}\right)\exp\left(-\frac{\int_A q_g(S)\mathrm{E}_{q_c^{i-}(Z_{i-}|S)} \log p(Z_{i-},S,Y)dS}{\int_A q_g(S)dS}\right) \\
\propto&\exp\left(\frac{\int_A q_g(S)\mathrm{E}_{q_c^{i-}(Z_{i-}|S)} \log p(X,Y)dS}{\int_A q_g(S)dS}\right),
\end{aligned}$$

we reach the conclusion in Theorem 4.2.1: for $A \in \mathcal{P}_i^g$, namely, for A such that $q_g(S)$ is not constantly 0 when $S \in A$, the optimal $\tilde{q}_c^{i,A}$ in (4.9) serves as the required optimiser in (4.8).

Uniqueness of the optimiser $\tilde{q}_c^i$ It is noted that such an optimiser $\tilde{q}_c^{i,A}(Z_i)$ in (4.46) or (4.9) is not necessarily unique for $A \in \mathcal{P}_i^g$. Specifically, from (4.42), each summand contributes a product of a non-negative constant $\int_A q_g(S)dS$ and the function inside the bracket to the value of the objective function $\mathrm{E}_{q_g}F(S,q_c)$. When this constant, $\int_A q_g(S)dS$, is a positive real number (not infinitesimal), a larger value of its multiplier (i.e. the function in the bracket) always results in a non-negligible, positive increase in $\mathrm{E}_{q_g}F(S,q_c)$.

As a result, all $q_c^{i,A}$ in (4.42) that are associated with positive $\int_A q_g(S)dS$ must be optimised as in (4.46) in order to maximise the objective function $\mathrm{E}_{q_g}F(S,q_c)$. This leads to the conclusion found in Remark 2 of Theorem 4.2.1: for $A \in \mathcal{P}_i$ such that $\int_A q_g(S)dS$ is positive (not infinitesimal), the optimiser $\tilde{q}_c^{i,A}(Z_i)$ in (4.9) is unique.

On the other hand, if the constant $\int_A q_g(S)dS$ ($A \in \mathcal{P}_i^g$) in (4.42) is infinitesimal, e.g. when A is a singleton set and $q_g(S)$ is positive at that value, the uniqueness of the optimiser $\tilde{q}_c^{i,A}(Z_i)$ becomes more complex. In such a scenario, varying values of the function inside the bracket, when multiplied by such an infinitesimal constant, seem to have a negligible effect on the objective function's value. This is especially true when compared to other summands in (4.42) that are associated with an non-negligible positive constant $\int_A q_g(S)dS$.

Therefore, if the cardinality of partition $\mathcal{P}_i$ is countable, which implies that the number of summands in (4.42) must also be countable, an arbitrary distribution may

be assigned to the optimal $\tilde{q}_c^{i,A}$ for $A \in \mathcal{P}_i$ such that $\int_A q_g(S)dS$ is zero or infinitesimal. This is because their impact on the ELBO's value is negligible when compared to other summands in (4.42) that are associated with a positive $\int_A q_g(S)dS$.

However, it is important to acknowledge that when there are infinite number of summands in (4.42), e.g. when q_c^i is *conditional everywhere*, the summands in (4.42) that may appear negligible could still significantly impact the value of $\mathrm{E}_{q_g} F(S, q_c)$ when combined. Consequently, a cautious approach for optimising the ELBO is to consistently employ the optimiser $\tilde{q}_c^{i,A}$ in (4.46) or (4.9) for $A \in \mathcal{P}_i^g$, although some of these optimisers may not be essential for specific partitions $\mathcal{P}_i$.

4.4.1.3 Derivation of the *conditional everywhere* optimiser $\tilde{q}_c^i$

We now derive the optimiser $\tilde{q}_c^i$ for *conditional everywhere* q_c^i and prove (4.10) on page 106. First note that $\mathrm{E}_{q_g} F(S, q_c)$ in (4.33) is

$$\mathrm{E}_{q_g} F(S, q_c) = \int_U q_g(S) F(S, q_c) dS. \tag{4.47}$$

$$F(S, q_c) = \mathrm{E}_{q_c^i(Z_i|S)} \left(\mathrm{E}_{q_c^{i-}(Z_{i-}|S)} \log p(Z_i|Z_{i-}, S, Y) - \log q_c^i(Z_i|S) \right). \tag{4.48}$$

Moreover, the *conditional everywhere* assumption allows $q_c^i(Z_i|S)$ to be determined independently for every different $S \in U$. Then for all $u \in U$ such that $q_g(S = u) = 0$, the parametric form of $q_c^i(Z_i|S = u)$ will not affect the value of $\mathrm{E}_{q_g} F(S, q_c)$ since the integrand in (4.47) will always be zero at those points. Therefore, $q_c^i(Z_i|S = u)$ can be any distribution for all $u \in U$ such that $q_g(S = u) = 0$, which agrees with Theorem 4.2.1.

Furthermore, when the maximum of $F(S, q_c)$ is achieved for all $S \in U$, by multiplying a non-negative constant $q_g(S)$, the integrand in (4.47) also achieves the maximum for every $S \in U$, and hence the objective function $\mathrm{E}_{q_g} F(S, q_c)$ in (4.47) is also maximised (since the integrand is maximised at every point). This suggests that one solution (not necessarily a unique one) for the *conditional everywhere* optimiser $\tilde{q}_c^i(Z_i|S) = \arg\max_{q_c^i} \mathrm{E}_{q_g} F(S, q_c)$ in (4.33) is

$$\tilde{q}_c^i(Z_i|S = u) = \underset{q_c^i(Z_i|S=u)}{\arg\max}\, F(S = u, q_c). \tag{4.49}$$

Now in order to rewrite the (4.48) with Lemma 4.4.1, we follow the similar procedure in (4.41) to check the following integral is finite:

$$\begin{aligned}
&\int \exp\left(\mathrm{E}_{q_c^{i-}(Z_{i-}|S)} \log p(Z_i|Z_{i-}, S, Y)\right) dZ_i \\
=&\mathrm{KL}(q_c^{i-}(Z_{i-}|S)||p(Z_{i-}|S, Y)) \int \exp\left(\mathrm{E}_{q_c^{i-}(Z_{i-}|S)} \log \frac{p(Z_i, Z_{i-}|S, Y)}{q_c^{i-}(Z_{i-}|S)}\right) dZ_i \\
\leq&\mathrm{KL}(q_c^{i-}(Z_{i-}|S)||p(Z_{i-}|S, Y)),
\end{aligned} \tag{4.50}$$

which is usually finite if q_c^{i-} are obtained following the standard routine of coordinate ascent update. Here we skip the detailed derivation of (4.50) and the discussion of the finiteness of the KL divergence in (4.50), since they are similar to those for (4.41). Now we can employ Lemma 4.4.1 to rewrite (4.48) as

$$\begin{aligned}
F(S, q_c) =&\mathrm{E}_{q_c^i(Z_i|S)} \left(\mathrm{E}_{q_c^{i-}(Z_{i-}|S)} \log p(Z_i|Z_{i-}, S, Y) - \log q_c^i(Z_i|S)\right) \\
=& -\mathrm{KL}(q_c^i(Z_i|S)||\tilde{p}_S(Z_i|S)) + c_S \qquad (4.51) \\
\tilde{p}_S(Z_i|S) \propto& \exp\left(\mathrm{E}_{q_c^{i-}(Z_{i-}|S)} \log p(Z_i|Z_{i-}, S, Y)\right), \qquad (4.52)
\end{aligned}$$

where c_S is a constant that does not depend on q_c or Z_i, and can be computed similarly as in (4.44). Now, by using (4.51) and (4.52), the optimiser $\tilde{q}_c^i$ in (4.49) is

$$\begin{aligned}
&\tilde{q}_c^i(Z_i|S = u) = \underset{q_c^i(Z_i|S=u)}{\arg\max}\, F(S = u, q_c) \\
=&\underset{q_c^i(Z_i|S=u)}{\arg\max} \left(-\mathrm{KL}(q_c^i(Z_i|S = u)||\tilde{p}_S(Z_i|S = u)) + c_S\right) \\
=&\tilde{p}_S(Z_i|S = u) \propto \exp\left(\mathrm{E}_{q_c^{i-}(Z_{i-}|S=u)} \log p(Z_i|Z_{i-}, S = u, Y)\right), \\
\propto& \exp\left(\mathrm{E}_{q_c^{i-}(Z_{i-}|S=u)} \log p(Z, S = u, Y)\right),
\end{aligned} \tag{4.53}$$

where the third line is obtained since $q_c^i(Z_i|S = u)$ can be chosen freely by the *conditional everywhere* assumption. The last line in (4.53) is obtained by multiplying the constant $\mathrm{E}_{q_c^{i-}} \log p(Z_{i-}, S = u, Y)$ to the third line. At last, by combining the result in (4.53), and our previous conclusion that $q_c^i(Z_i|S = u)$ can be any distribution for all $u \in U$ such that $q_g(S = u) = 0$, we obtain the *conditional everywhere* optimiser stated in Theorem 4.2.1.

4.4.2 Detailed derivation of the approximate local update

Here, we show the details how the importance sampling is employed in the local update in (4.13) for approximating (4.9) on page 106. First, we rewrite the optimal update for $\tilde{q}_c^{i,A}(Z_i)$ in (4.9) as follows

$$\begin{aligned}\tilde{q}_c^{i,A}(Z_i) &\propto \exp\left(\frac{\int_A q_g(S)\mathrm{E}_{q_c^{i-}(Z_{i-}|S)}\log p(X,Y)dS}{\int_A q_g(S)dS}\right)\\ &= \exp\left(\frac{\int_U I(S\in A)q_g(S)\mathrm{E}_{q_c^{i-}(Z_{i-}|S)}\log p(X,Y)dS}{\int_U I(S\in A)q_g(S)dS}\right)\\ &= \exp\left(\frac{\int_U \lambda(S)I(S\in A)\frac{Cq_g(S)}{\lambda(S)}\mathrm{E}_{q_c^{i-}(Z_{i-}|S)}\log p(X,Y)dS}{\int_U \lambda(S)I(S\in A)\frac{Cq_g(S)}{\lambda(S)}dS}\right),\end{aligned} \tag{4.54}$$

where C can be an arbitrary constant. Recall that this optimal iterative update for $\tilde{q}_c^{i,A}(Z_i)$ in Algorithm 7 is carried out with the recently updated q_g in (4.7), in which case we have $q_g(S) \propto \tilde{\omega}(S)\lambda(S)$, with the unnormalised weight $\tilde{\omega}(S)$ being defined in (4.11). We can chose the C in (4.54) as the normalisation constant such that $\frac{Cq_g(S)}{\lambda(S)} = \tilde{\omega}(S)$. Subsequently, the optimal update for $\tilde{q}_c^{i,A}(Z_i)$ in (4.54) is

$$\tilde{q}_c^{i,A}(Z_i) \propto \exp\left(\frac{\int_U \lambda(S)I(S\in A)\tilde{\omega}(S)\mathrm{E}_{q_c^{i-}(Z_{i-}|S)}\log p(X,Y)dS}{\int_U \lambda(S)I(S\in A)\tilde{\omega}(S)dS}\right). \tag{4.55}$$

Now, with N_p particles $S^{(p)}$ ($p \in \mathcal{I} = \{1,2,...,N_p\}$) independently drawn from the proposal $\lambda(S)$, the numerator and denominator inside the exponent in (4.55) can be respectively approximated with Monte Carlo methods as follows,

$$\begin{aligned}&\int_U \lambda(S)I(S\in A)\tilde{\omega}(S)\mathrm{E}_{q_c^{i-}(Z_{i-}|S)}\log p(X,Y)dS\\ \approx&\frac{1}{N_p}\sum_{p=1}^{N_p} I(S^{(p)}\in A)\tilde{\omega}^{(p)}\mathrm{E}_{q_c^{i-}(Z_{i-}|S^{(p)})}\log p(S^{(p)},Z,Y)\\ =&\frac{1}{N_p}\sum_{j\in B}\tilde{\omega}^{(j)}\mathrm{E}_{q_c^{i-}(Z_{i-}|S^{(j)})}\log p(S^{(j)},Z,Y),\end{aligned} \tag{4.56}$$

$$\begin{aligned}\int_U \lambda(S)I(S\in A)\tilde{\omega}(S)dS &\approx \sum_{p=1}^{N_p} I(S^{(p)}\in A)\tilde{\omega}^{(j)}\\ &= \frac{1}{N_p}\sum_{j\in B}\tilde{\omega}^{(j)}\end{aligned} \tag{4.57}$$

where $B = \{p \in \mathcal{I} : S^{(p)} \in A\}$ and $\tilde{\omega}^{(p)}$ is given in (4.12). Naturally, these approximations require that $B \neq \emptyset$. Substituting these two approximations into (4.55) yields our approximated update $\hat{q}_c^{i,B}$ in (4.13).

4.4.3 Proof of Theorem 4.3.1

Here we present the proof for Theorem 4.3.1 on page 111. Note from (4.28) that

$$\begin{aligned}\mathcal{F}\left(\tilde{q}_g, q_c^1, q_c^2, ..., q_c^{N_c}\right) =& -\mathrm{KL}(\tilde{q}_g(S)||\tilde{p}(S)) + c + \log p(Y) \\ =& c + \log p(Y),\end{aligned} \tag{4.58}$$

where the last equality follows since $\tilde{q}_g(S)$ is the exact optimal global distribution computed in (4.7) as stated in Theorem 4.3.1, and we have demonstrated that $\tilde{q}_g(S) = \tilde{p}(S)$ in (4.31). The constant c can be computed according to (4.30):

$$\exp(c) = \int_U \frac{\exp\left(\mathrm{E}_{\prod_{i=1}^{N_c} q_c^i(Z_i|S)} \log p(X|Y)\right)}{\prod_{i=1}^{N_c} \exp\left(\mathrm{E}_{q_c^i(Z_i|S)} \log q_c^i(Z_i|S)\right)} dS, \tag{4.59}$$

where U is the domain of S. By using (4.15) and (4.12) on pages 111 and 109, respectively, we have

$$\begin{aligned}&\mathrm{E}_{\prod_{p=1}^{N_p} \lambda(S^{(p)})} \exp\left(\hat{\mathcal{F}}\right) \\ =&\mathrm{E}_{\prod_{p=1}^{N_p} \lambda(S^{(p)})} \frac{1}{N_p} \sum_{p=1}^{N_p} \tilde{\omega}^{(p)} = \mathrm{E}_{\lambda(S^{(1)})} \tilde{\omega}^{(1)} \\ =&\mathrm{E}_{\lambda(S^{(1)})} \frac{\exp\left(\mathrm{E}_{\prod_{i=1}^{N_c} q_c^i(Z_i|S^{(1)})} \log p(Z, S^{(1)}, Y)\right)}{\lambda(S^{(1)}) \prod_{i=1}^{N_c} \exp\left(\mathrm{E}_{q_c^i(Z_i|S^{(1)})} \log q_c^i(Z_i|S^{(1)})\right)} \\ =&\int_U \frac{\exp\left(\mathrm{E}_{\prod_{i=1}^{N_c} q_c^i(Z_i|S^{(1)})} \log p(Z, S^{(1)}, Y)\right)}{\prod_{i=1}^{N_c} \exp\left(\mathrm{E}_{q_c^i(Z_i|S^{(1)})} \log q_c^i(Z_i|S^{(1)})\right)} dS^{(1)} \\ =&\exp(c) p(Y) = \exp\left(\mathcal{F}\left(\tilde{q}_g, q_c^1, q_c^2, ..., q_c^{N_c}\right)\right),\end{aligned} \tag{4.60}$$

where the last line is obtained by using (4.59) and (4.58). Subsequently, by using Jensen's inequality and (4.60) we have

$$\begin{aligned}\mathrm{E}_{\prod_{p=1}^{N_p}\lambda(S^{(p)})}\left(\hat{\mathcal{F}}\right) \leq & \log\left(\mathrm{E}_{\prod_{p=1}^{N_p}\lambda(S^{(p)})}\exp\left(\hat{\mathcal{F}}\right)\right)\\ = & \mathcal{F}\left(\tilde{q}_g, q_c^1, q_c^2, ..., q_c^{N_c}\right).\end{aligned}$$

This inequality and (4.60) together fulfill both results outlined in Theorem 4.3.1, thus completing the proof.

4.4.4 Proof of Theorem 4.3.2

Here we prove Theorem 4.3.2 on page 112. Note that the IS-CVB described in Algorithom (8) is deterministic since they are recursively carried out with the same previously drawn samples $S^{(p)}(p \in \mathcal{I})$. We will first show that 1) such deterministic updates essentially perform a deterministic theoretic CVB (Algorithm 7) for a modified target distribution (denoted as $\check{p}(\cdot)$), for which a modified ELBO (denoted as $\mathcal{L}$) is guaranteed to increase across iterations; and 2) the estimated ELBO $\hat{\mathcal{F}}$ computed in Algorithm 8 is equal to this modified ELBO $\mathcal{L}$ up to a constant. These two facts imply that the sequence $(\hat{\mathcal{F}}_k)_{k\in\mathbb{N}}$ is also monotonically increasing. Finally, we show the convergence of $(\hat{\mathcal{F}}_k)_{k\in\mathbb{N}}$ by verifying $\hat{\mathcal{F}}$ is bounded above.

We now prove the first part. Define $J \in \mathcal{I} = \{1, 2, ..., N_p\}$ as a discrete random variable, and a modified joint target distribution $\check{p}$ for variables J, Z, Y satisfying

$$\check{p}(J, Z, Y) \propto \frac{p(S^{(J)}, Z, Y)}{\lambda(S^{(J)})}, \tag{4.61}$$

where variables S, Z, Y and the exact distribution p have the same definition as in Section 4.2.1; $S^{(J)}$ is the J-th previously drawn sample (from the proposal λ) used in Algorithm 8. This definition in (4.61) suggests that the modified joint distribution can be exactly evaluated as

$$\check{p}(J, Z, Y) = \frac{p(S^{(J)}, Z, Y)}{\lambda(S^{(J)})}\left(\sum_{p=1}^{N_p}\frac{p(S^{(p)})}{\lambda(S^{(p)})}\right)^{-1}. \tag{4.62}$$

We will approximate $\check{p}(J, Z|Y)$ by the following conditionally factorised variational distribution $r(J, Z)$:

$$\begin{aligned} r(J, Z) =& r_g(J) \prod_{i=1}^{N_c} r_c^i(Z_i|J), \\ r_c^i(Z_i|J) =& \sum_{B \in \mathcal{S}_i} r_c^{i,B}(Z_i) I(J \in B), \end{aligned} \tag{4.63}$$

for $i = 1, 2, ..., N_c$, where r_c^i is assumed to be *setwise conditional* on the partition $\mathcal{S}_i$. Note that we assume the partitions $\mathcal{S}_i$ $(i = 1, 2, ..., N_c)$ are the same particle index partitions used in Algorithm 8. We now define the following modified ELBO $\mathcal{L}$:

$$\mathcal{L} = \mathrm{E}_{r_g(J) \prod_{i=1}^{N_c} r_c^i(Z_i|J)} \log \frac{\check{p}(J, Z, Y)}{r_g(J) \prod_{i=1}^{N_c} r_c^i(Z_i|J)}, \tag{4.64}$$

where we can see $\mathcal{L}$ is a standard ELBO similar to (4.2) where the exact distribution and variational distribution are now $\check{p}$ and r. The optimisation of this modified ELBO with respect to the conditionally factorised variational family $\{r\}_{cf}$ in (4.63) can be carried out with our theoretical CVB (Algorithm 7). We now derive coordinate ascent optimisers according to Theorem 4.2.1.

Specifically, the optimal variational updates for r_g in (4.7) is:

$$\begin{aligned} \tilde{r}_g(J) &\propto \exp\left(\mathrm{E}_{\prod_{i=1}^{N_c} r_c^i(Z_i|J)} \log \frac{\check{p}(J, Z, Y)}{\prod_{i=1}^{N_c} r_c^i(Z_i|J)} \right) \\ &= \exp\left(\mathrm{E}_{\prod_{i=1}^{N_c} r_c^i(Z_i|J)} \log \frac{p(S^{(J)}, Z, Y)}{\lambda(S^{(J)}) \prod_{i=1}^{N_c} r_c^i(Z_i|J)} - \log \sum_{p=1}^{N_p} \frac{p(S^{(p)})}{\lambda(S^{(p)})} \right) \\ &\propto \exp\left(\mathrm{E}_{\prod_{i=1}^{N_c} r_c^i(Z_i|J)} \log \frac{p(S^{(J)}, Z, Y)}{\lambda(S^{(J)}) \prod_{i=1}^{N_c} r_c^i(Z_i|J)} \right) \\ &= \frac{\exp\left(\mathrm{E}_{\prod_{i=1}^{N_c} r_c^i(Z_i|J)} \log p(S^{(J)}, Z, Y) \right)}{\lambda(S^{(J)}) \prod_{i=1}^{N_c} \exp\left(\mathrm{E}_{r_c^i(Z_i|J)} \log r_c^i(Z_i|J) \right)} = f(J), \end{aligned} \tag{4.65}$$

where the second line is obtained by using (4.62); and we define $f(J)$ as

$$f(J) = \frac{\exp\left(\mathrm{E}_{\prod_{i=1}^{N_c} r_c^i(Z_i|J)} \log p(S^{(J)}, Z, Y) \right)}{\lambda(S^{(J)}) \prod_{i=1}^{N_c} \exp\left(\mathrm{E}_{r_c^i(Z_i|J)} \log r_c^i(Z_i|J) \right)}. \tag{4.66}$$

Note that (4.65) also suggests that $\tilde{r}_g(J)$ can be computed as follows

$$\tilde{r}_g(J) = \frac{f(J)}{\sum_{J=1}^{N_p} f(J)}. \tag{4.67}$$

According to Theorem 4.2.1 and (4.9), the optimal distribution for r_c^i can be expressed as follows,

$$\tilde{r}_c^i(Z_i|J) = \sum_{B \in \mathcal{S}_i} \tilde{r}_c^{i,B}(Z_i) I(J \in B),$$

where $\tilde{r}_c^{i,B}(Z_i)$ can be any distribution if $\sum_{J\in B} r_g(J) = 0$; otherwise $\tilde{r}_c^{i,B}(Z_i)$ is

$$\begin{aligned}
&\tilde{r}_c^{i,B}(Z_i) \propto \exp\left(\frac{\sum_{J\in B} r_g(J) \mathrm{E}_{r_c^{i-}(Z_{i-}|J)} \log \check{p}(J, Z, Y)}{\sum_{J\in B} r_g(J)}\right) \\
=&\exp\left(\frac{\sum_{J\in B} r_g(J) \mathrm{E}_{r_c^{i-}(Z_{i-}|J)} \log p(S^{(J)}, Z, Y)}{\sum_{J\in B} r_g(J)} - \frac{\sum_{J\in B} r_g(J) \log \lambda(S^{(J)}) \sum_{p=1}^{N_p} \frac{p(S^{(p)})}{\lambda(S^{(p)})}}{\sum_{J\in B} r_g(J)}\right) \\
\propto&\exp\left(\frac{\sum_{J\in B} r_g(J) \mathrm{E}_{r_c^{i-}(Z_{i-}|J)} \log p(S^{(J)}, Z, Y)}{\sum_{J\in B} r_g(J)}\right),
\end{aligned} \tag{4.68}$$

where the second line is obtained by substituting (4.62). Compare these updates (4.65) and (4.68) required in the theoretical CVB (Algorithm 7) for such a modified variational inference task with the Monte Carlo approximated updates (4.12) and (4.13) required in the IS-CVB (Algorithm 8), we can find the following relationship always holds

$$\begin{aligned}
f(J) &= \tilde{\omega}^{(J)}, \\
r_g(J) &= \omega^{*(J)}, \\
r_c^{i,B}(Z_i) &= \hat{q}_c^{i,B}(Z_i), \\
r_c^i(Z_i|J) &= q_c^i(Z_i|S^{(J)}),
\end{aligned} \tag{4.69}$$

if 1) the initial distributions of $\{r\}_{cf}$ for starting Algorithm 7 and the initial distributions of $\{q\}_{cf}$ for starting Algorithm 8 are the same; 2) the same refinement steps for $\mathcal{S}_i$ are of carried out for each algorithm; and 3) $\tilde{r}_c^{i,B}(Z_i)$ and $\hat{q}_c^{i,B}$ are set to the same distribution when $\sum_{j\in B} \tilde{\omega}^{(j)} = 0$ and $\sum_{J\in B} r_g(J) = 0$. This shows that the IS-CVB in Algorithm 8 essentially performs the theoretical CVB for the modified variational inference task described above. In particular, the refinement of $\mathcal{S}_i$ in IS-CVB (Algorithm 8) corresponds to the refinement of the partitions of J's domain in the theoretical CVB

(Algorithm 7) for the modified variational inference task. The weight evaluation in (3.47) and approximate local update (4.13) in IS-CVB are essentially the coordinate ascent updates (4.65) and (4.68) in theoretical CVB. As discussed in Section 4.2.2, such a theoretical CVB guarantees the increase of the ELBO for this modified variational inference task. Therefore, each step of $\mathcal{S}_i$ refinement, weight evaluation, local update in the IS-CVB (Algorithm 8) must also guarantee a non-negative increment of the modified ELBO $\mathcal{L}$ in (4.64).

Next, we show the second part of the proof. When the global update in (4.65) is just carried out with the latest update r_c, i.e. $\{r\}_{cf} = \{\tilde{r}_g, r_c^1, r_c^2, ...r_c^{N_c}\}$, we have

$$
\begin{aligned}
\mathcal{L} =& \mathrm{E}_{\tilde{r}_g(J)\prod_{i=1}^{N_c} r_c^i(Z_i|J)} \log \frac{\check{p}(J,Z,Y)}{\tilde{r}_g(J)\prod_{i=1}^{N_c} r_c^i(Z_i|J)} \\
=& \mathrm{E}_{\tilde{r}_g(J)\prod_{i=1}^{N_c} r_c^i(Z_i|J)} \log \frac{p(S^{(J)},Z,Y)}{\lambda(S^{(J)})\tilde{r}_g(J)\prod_{i=1}^{N_c} r_c^i(Z_i|J)} - \log \sum_{p=1}^{N_p} \frac{p(S^{(p)})}{\lambda(S^{(p)})} \\
=& \mathrm{E}_{\tilde{r}_g(J)} \left(\log f(J) - \log \tilde{r}_g(J)\right) - \log \sum_{p=1}^{N_p} \frac{p(S^{(p)})}{\lambda(S^{(p)})} \\
=& \mathrm{E}_{\tilde{r}_g(J)} \log \sum_{J=1}^{N_p} f(J) - \log \sum_{p=1}^{N_p} \frac{p(S^{(p)})}{\lambda(S^{(p)})} \\
=& \log \sum_{J=1}^{N_p} f(J) - \log \sum_{p=1}^{N_p} \frac{p(S^{(p)})}{\lambda(S^{(p)})} \\
=& \log \sum_{p=1}^{N_p} \tilde{\omega}^{(p)} - \log \sum_{p=1}^{N_p} \frac{p(S^{(p)})}{\lambda(S^{(p)})} \\
=& \hat{\mathcal{F}} - \log N_p - \log \sum_{p=1}^{N_p} \frac{p(S^{(p)})}{\lambda(S^{(p)})},
\end{aligned}
\tag{4.70}
$$

where the second, third and fourth lines are obtained by using (4.62), (4.65), and (4.67) respectively. The fifth line is obtained by using the fact that $\log \sum_{J=1}^{N_p} f(J)$ is a constant that no longer depends on J. The sixth, and the last lines are obtained by substituting (4.69), and (4.15) respectively. (4.70) shows that $\mathcal{L}$ and $\hat{\mathcal{F}}$ are equal up to an additive constant.

Moreover, we can check $\hat{\mathcal{F}}$ is always bounded above. Specifically, by using the last line and the second line in (4.70), we have

$$\begin{aligned}\hat{\mathcal{F}} =& \mathrm{E}_{\tilde{r}_g(J)\prod_{i=1}^{N_c} r_c^i(Z_i|J)} \log \frac{p(S^{(J)}, Z, Y)}{\lambda(S^{(J)})\tilde{r}_g(J)\prod_{i=1}^{N_c} r_c^i(Z_i|J)} + \log N_p \\ \leq& \log \mathrm{E}_{\tilde{r}_g(J)\prod_{i=1}^{N_c} r_c^i(Z_i|J)} \frac{p(S^{(J)}, Z, Y)}{\lambda(S^{(J)})\tilde{r}_g(J)\prod_{i=1}^{N_c} r_c^i(Z_i|J)} + \log N_p \\ =& \log \sum_{J=1}^{N_p} \frac{p(S^{(J)}, Y)}{\lambda(S^{(J)})} + \log N_p,\end{aligned} \tag{4.71}$$

where the second line is obtained using Jensen's inequality. This shows that $\hat{\mathcal{F}}$ is always bounded above by a constant in the last line of (4.71).

Finally we show the convergence of $(\hat{\mathcal{F}}_k)_{k\in\mathbb{N}}$. Recall that each $(\hat{\mathcal{F}}_k)$ is evaluated in Algorithm 8 for each iteration after carrying out 1) partition refinement, 2) approximate local update, and 3) weight evaluation. We have showed that all these three steps in Algorithm 8 guarantee a non-negative increment of the modified ELBO $\mathcal{L}$ in (4.64). Then because (4.70) shows that $\mathcal{L}$ equals $\hat{\mathcal{F}}$ plus a constant, the sequence $(\hat{\mathcal{F}}_k)_{k\in\mathbb{N}}$ must also be monotonically increasing across the iterative updates in Algorithm 8. Note that the refinement step in the theoretical CVB, which can be carried out manually at any desired iteration number, may lead to further updates with a non-negative increment of $\mathcal{L}$; and henceforth the $\mathcal{F}_k$ can potentially increase at any desired iteration number. Therefore, the index partitions $\mathcal{S}_i$ $(i = 1, 2, ..., N_c)$ need to be fixed after some iteration number to ensure $(\hat{\mathcal{F}}_k)_{k\in\mathbb{N}}$ is a standard sequence without manual regulation. The convergence of such a sequence is then guaranteed by the fact that $(\hat{\mathcal{F}}_k)_{k\in\mathbb{N}}$ is a sequence of real numbers, and it is monotonically increasing and bounded above as shown in (4.71).

4.5 Simulation Result

In this section, we verify the proposed IS-CVB and Theorem 4.3.2 on a simple example. Consider the following Bayesian model:

$$p(\mu, \nu) = \mathcal{N}(\mu; 0, 2)\mathcal{G}(\nu; 2, 0.5),$$

$$p(x|\mu, \nu) = \mathcal{N}\left(x; \mu, \nu^{-1}\right), \quad p(y|x) = \mathcal{N}\left(y; 0.5x^2, 1 + e^{10x}\right),$$

where $\mathcal{G}(k, \theta)$ is the gamma distribution with shape parameter k and scale parameter θ. We are interested in the following two posteriors: 1) $p(x, \mu, \nu|y = 8)$, and 2) $p(x, \mu, \nu|y = 25)$. As the exact posteriors are analytically intractable, we approximate them within the proposed CVB framework. We consider the conditionally factorised families which satisfy the factorisation $q(x, \mu, \nu) = q_g(x)q_c^\mu(\mu|x)q_c^\nu(\nu|x)$, and adopt the following five different partitions $\mathcal{P}_\mu, \mathcal{P}_\nu$ for q_c:
(i) fully factorised: $\mathcal{P}_\mu = \{\mathbb{R}\}, \mathcal{P}_\nu = \{\mathbb{R}\}$;
(ii) jointly factorised 1: *conditional everywhere* q_c^μ and $\mathcal{P}_\nu = \{\mathbb{R}\}$;
(iii) jointly factorised 2: $\mathcal{P}_\mu = \{\mathbb{R}\}$ and *conditional everywhere* q_c^ν;
(iv) setwise conditional: $\mathcal{P}_\mu = \mathcal{P}_\nu = \{\mathbb{R}_{\geq 0}, \mathbb{R}_{<0}\}$;
(v) conditional everywhere: *conditional everywhere* q_c^μ and q_c^ν.
Note that the conditionally factorised families with settings (i)-(iii) degenerate to the mean-field families. The above five settings are implemented in IS-CVB without refinement steps, i.e. the partitions are fixed all the time. We also consider another two settings:
(vi): initialised with (i), switches to (iii), then to (v) if converged;
(vii): initialised with (i), switches to (iv), then to (v) if converged,
where both are initialised with the setting (i), and then switch to a setting with *finer* partitions twice if the algorithm has converged.

All settings (i)-(vii) are implemented by using IS-CVB with the same particle set ($N_p = 1000$) and the same initialised distributions. The proposal we use is $\lambda(x) = 0.3\mathcal{N}(x; m_1, c_1) + 0.7\mathcal{N}(x; 1, 1.5)$, where $m_1 = -\sqrt{2y}, c_1 = 1 + e^{10m_1}$. This proposal is designed to cover the possible range of x that can generate the given observation y. In particular, if $x \leq -1$, the true x is very likely to be covered by $\mathcal{N}(x; m_1, c_1)$; and the $\mathcal{N}(x; 1, 1.5)$ should easily cover the region $-1 < x \leq 3$; and we do not consider covering the region $x > 3$, as in such a case the posterior $p(x|y) \propto p(y|x)p(x)$ is very low no matter what y is received. The implementation of IS-CVB with the settings (i)-(iii) are simply Monte Carlo based mean-field CAVI, e.g. in (ii) each sample of x are associated with a closed-form $q(\mu|x)$; and the standard CAVI is analytically intractable for this model.

To derive the corresponding update for IS-CVB, first note that $p(\mu|\nu, y)$ is a Gaussian density with argument μ and $p(\nu|\mu, y)$ is a gamma density with argument of ν. Then according to Section 4.3.3, the parametric form of $\hat{q}_c^{\mu,B}(\mu)$ and $\hat{q}_c^{\nu,B}(\nu)$ in our local updates are Gaussian and gamma density respectively. It can be showed that for

$B \in \mathcal{P}_\mu$, the local update $\hat{q}_c^{\mu,B}(\mu)$ in (4.13) yields

$$\begin{aligned}\hat{q}_c^{\mu,B}(\mu) &= \mathcal{N}\left(\mu; \frac{C^{-1}m + \sum_{p\in B}\omega_B^{*(p)}\overline{\nu}^{(p)}x^{(p)}}{C^{-1} + \sum_{p\in B}\omega_B^{*(p)}\overline{\nu}^{(p)}}, \frac{1}{C^{-1} + \sum_{p\in B}\omega_B^{*(p)}\overline{\nu}^{(p)}}\right), \\ \omega_B^{*(p)} &= \frac{\tilde{\omega}^{(p)}}{\sum_{p\in B}\tilde{\omega}^{(p)}},\end{aligned} \tag{4.72}$$

where m, C are the mean and variance of $p(\mu)$ (0 and 2 in this example), and $\overline{\nu}^{(p)} = \mathrm{E}_{q(\nu|x^{(p)})}\nu$; and for $B \in \mathcal{P}_\nu$, the $\hat{q}_c^{\mu,B}(\mu)$ is

$$\begin{aligned}\hat{q}_c^{\mu,B}(\mu) &= \mathcal{G}\left(\nu; k+0.5, (\frac{1}{\theta} + \frac{\sum_{p\in B}\omega_B^{*(p)}\left((x^{(p)} - \overline{\mu}^{(p)})^2 + \mathrm{Var}[\mu]^{(p)}\right)}{2})^{-1}\right), \\ \omega_B^{*(p)} &= \frac{\tilde{\omega}^{(p)}}{\sum_{p\in B}\tilde{\omega}^{(p)}},\end{aligned} \tag{4.73}$$

where k and θ are the shape and scale parameter in $p(\nu)$ (2 and 0.5 in this example) and $\overline{\mu}^{(p)}$, $\mathrm{Var}[\mu]^{(p)}$ are the mean and variance of μ with respect to $q(\mu|x^{(p)})$.

The *conditional everywhere* update and fully factorised update can be obtained from (4.72) and (4.73) by setting $\mathcal{P}_\mu$ and $\mathcal{P}_\nu$ to the corresponding partitions discussed in the Remark 2 for Definition 4.2.1. The unnormalised weight $\tilde{\omega}^{(p)}$ can be evaluated as follows according to (4.12): denote $q(\nu|x^{(p)}) = \mathcal{G}(a^{(p)}, b^{(p)})$, then

$$\begin{aligned}\tilde{\omega}^{(p)} &= \exp\left(\mathrm{E}_{q(\mu,\nu|x^{(p)})}\log\frac{p(y,x^{(p)},\mu,\nu)}{q(\mu,\nu|x^{(p)})} - \log(\lambda(x^{(p)}))\right), \\ \mathrm{E}_{q(\mu,\nu|x^{(p)})}\log\frac{p(y,x^{(p)},\mu,\nu)}{q(\mu,\nu|x^{(p)})} &= (k-0.5)(\psi(a^{(p)}) + \log(b^{(p)})) - \frac{a^{(p)}b^{(p)}}{\theta} + a^{(p)} + \log(b^{(p)}) \\ &\quad - 0.5C^{-1}(\mathrm{Var}[\mu]^{(p)} + (\overline{\mu}^{(p)} - m)^2) - 0.5a^{(p)}b^{(p)}\mathrm{Var}[\mu]^{(p)} + \log(\Gamma(a^{(p)})) \\ &\quad + (1-a^{(p)})\psi(a^{(p)}) + 0.5\log(\mathrm{Var}[\mu]^{(p)}) + h(x^{(p)}) + const_p, \\ h(x) &= -0.5a^{(p)}b^{(p)}(x-\overline{\mu})^2 - 0.5\log(1+e^{tx}) - 0.5(1+e^{tx})^{-1}(y-0.5x^2)^2, \\ const_p &= -\log\Gamma(k) - k\log\theta - 0.5\log(C) - \log(2\pi) + 0.5,\end{aligned} \tag{4.74}$$

where $const_p$ is the constant that does not depend on p and hence can be omitted in the unnormalised weight.

The IS-CVB is implemented for all settings (i)-(vii) until convergence. The marginal variational distribution $q(\mu)$ and $q(\nu)$ for all the settings are shown in Fig. 4.1, except (vi) and (vii) as they are highly overlapped with setting (v). We can see that the proposed conditionally factorised families (iv)-(v) effectively capture the multimodal

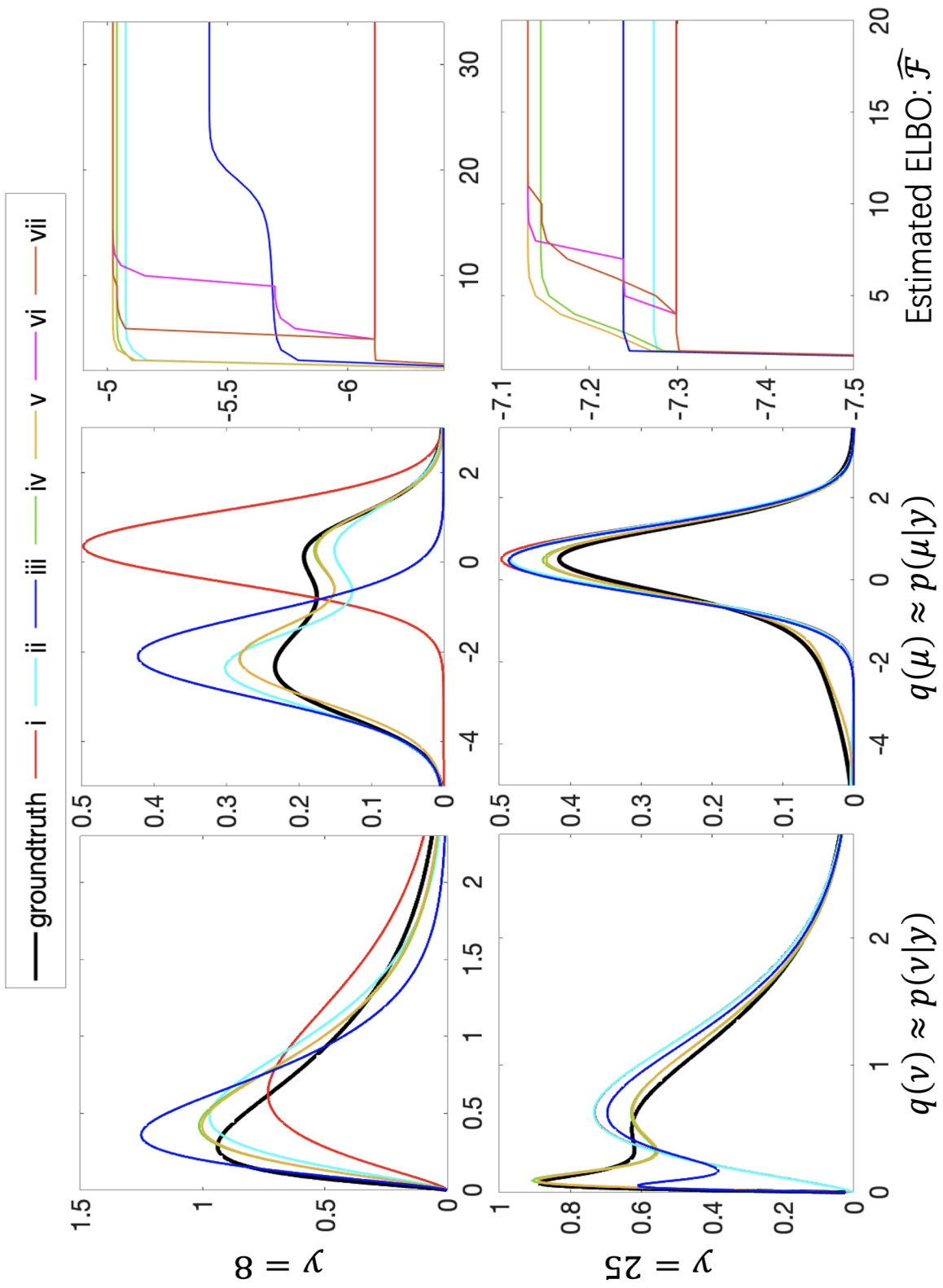


Fig. 4.1 Marginal variational distributions and the estimated ELBO. The groundtruth is obtained using a numerical grid-based method.

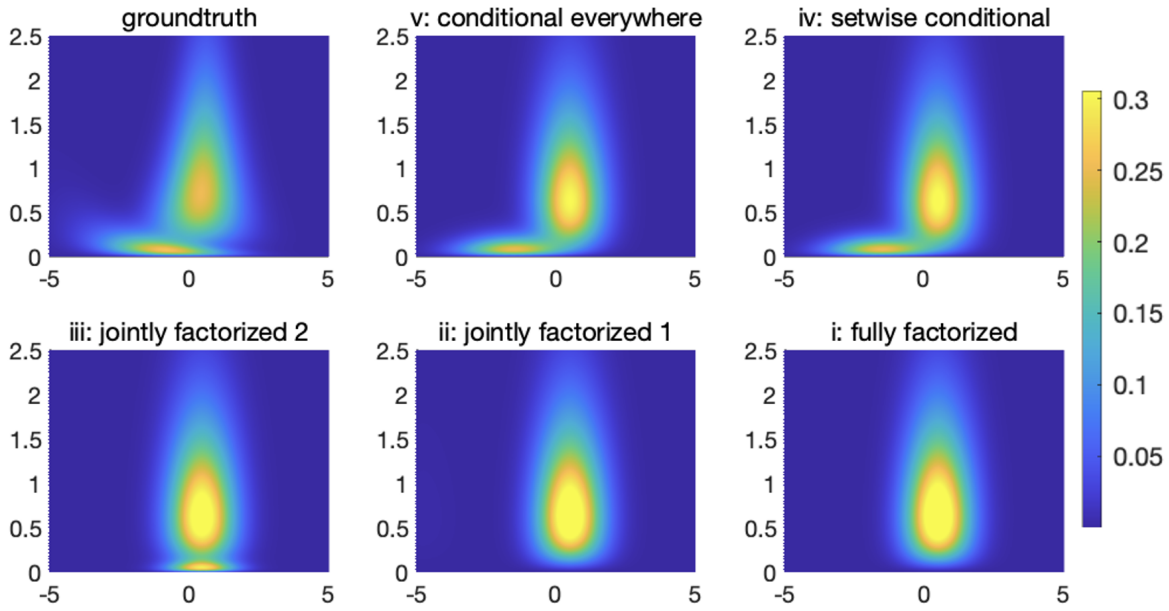


Fig. 4.2 Joint variational distributions $q(\mu, \nu) \approx p(\mu, \nu | y = 25)$. The groundtruth is obtained using a numerical grid-based method.

behaviour, and produce better fitted marginal approximations than standard mean-field settings (i)-(iii) for all considered distributions. The advantage of our method is further supported by their higher estimated ELBO $\hat{\mathcal{F}}$ in Fig. 4.1. The stepped curve of the $\hat{\mathcal{F}}$ produced by setting (vi) and (vii) satisfies Theorem 4.3.2, and guarantees a higher $\hat{\mathcal{F}}$ produced with the proposed CVB method compared to standard Monte Carlo mean-field CAVI.

Furthermore, the joint variational distributions $q(\mu, \nu)$ which approximate the posterior with observation $y = 25$ (evaluated via (4.14)) are plotted in Fig. 4.2, where the dependence between μ and ν is clearly retained in the proposed conditionally factorised family, i.e. settings (iv) and (v), and in contrast lost in other mean-field settings, i.e. (i)-(iii). Finally from Fig. 4.1 and Fig. 4.2, the *setwise conditional* setting (iv) achieves a competitive performance as the most accurate conditional *conditional everywhere* setting. We emphasise that it only requires the evaluation of 4 *free-form* optimal variational distributions (two for each q_c) in the implementation of IS-CVB, which is even less than the jointly factorised mean-field cases (ii) and (iii) where $N_p + 1$ *free-form* variational distributions need to be determined.

4.6 Conclusion

This chapter presents a generic variational family that can account for the dependence between variables with user selected detail. The resulting CVB algorithm offers a flexible trade-off between inference accuracy and computational cost. The theoretical CVB and its importance sampling implementation IS-CVB are derived. We prove several useful properties of the algorithm such as the guaranteed performance improvement compared to the standard mean-field CAVI, and the monotonically increasing property of the estimated ELBO. The tractability of the resulting update and the applicability of the algorithm are also discussed. Although the algorithm is only demonstrated here with a simple posterior computation task, it forms the basis for a sequential implementation that features the importance sampling particle filter. We have also developed it into a full filtering and online parameter estimation strategy for time series in a conditionally Gaussian system (e.g. Lévy state-space models in Chapter 3 and [91, 142, 69, 94]), results of which will be reported in future.

Chapter 5

A Variational Bayes Association-based Multi-object Tracker under the Non-homogeneous Poisson Measurement Process

In the object tracking applications discussed in chapters 2 and 3, it is assumed that we know which measurements originate from which objects. However, in many real-world applications, sensors cannot distinguish between different objects. Consequently, we only receive data that includes measurements from all objects in the surveillance area, as well as clutter resulting from obstacles or environmental noise, without knowing which data are associated with specific objects. To address this challenge, the multi-object tracking problem in such a scenario will be the focus of this and the following chapter.

The non-homogeneous Poisson process (NHPP) has been widely used to model extended object measurements where one object can generate zero or several measurements; it also provides an elegant solution to the computationally demanding data association problem in multiple object tracking. This chapter presents a variational Bayes association-based NHPP tracker (VB-AbNHPP) that can efficiently perform tracking, data association, and learning of target and clutter rates with a parallelisable implementation. This VB-AbNHPP tracker can be easily extended to online learn other static parameter such as measurement covariance/object extent based on a general

coordinate ascent variational filtering framework developed here. The results show that the proposed VB-AbNHPP tracker is superior to other competing methods in terms of implementation efficiency and tracking accuracy. The results in this chapter partly appeared in a prior publication, [143][1].

5.1 Introduction

The NHPP tracker provides an exact measurement likelihood and avoids the combinatorial complexity of the data association problem [144]. However, the original scheme in [144] is implemented by using a particle filter (PF), which could be computationally expensive for practical application; it also suffers from 'the curse of dimensionality' of particle filtering. In contrast, sequential Markov chain Monte Carlo (SMCMC) has proven to be a promising methodology to handle high-dimensional problems [145]. Therefore, in [146], a sequential MCMC scheme was utilised to infer online the target state and association variables under the NHPP measurement model. With a large enough sample size, this method can theoretically converge to an optimal Bayesian filter.

Complementary to the aforementioned numerical sampling methods, deterministic approximation methods typically feature a fast implementation, which makes them appealing in practical applications. One example is the extended target Joint Probabilistic Data Association (ET-JPDA) filter proposed in [147]. It adopts the same NHPP measurement model but falsely assumes an independent predictive likelihood for each measurement conditional on associations; hence, the derived independent marginal association posterior in [147] is regarded as a rough approximation. Moreover, the moment matching strategy in the target state update also introduces approximation. Particularly, for each target, it recursively updates the target state with one single measurement by moment matching until exhausting all measurements within the gate. Such procedures also raise a concern that the order of the measurements may affect the update result. Approximate inference techniques, including graphical model methods, have been employed to tackle the data association problem [148, 149]. However, the approaches presented in [148, 149] are specifically tailored for point target measurement models, rendering them unsuitable for NHPP measurement models. Although an extension of the message-passing algorithm [150] accommodates NHPP model, its reliance on a particle filter leads to significant computational inefficiency, particularly in scenarios with massive measurement data demonstrated in [50]. Another tracking method that

[1] © 2022 IEEE. Reprinted, with permission, from [143]

adopts an approximate inference method is the probabilistic multiple hypothesis tracker (PMHT) in [151], of which measurement model is inherently equivalent to the NHPP model conditional on a known measurement number. However, the PMHT method utilises a batch expectation maximisation (EM) technique and thus could not track targets in an online fashion.

5.1.1 Contributions

In this chapter, we propose a variational Bayes association based NHPP tracker that can estimate online the target kinematics and the association variables in parallel. It provides a promising alternative to the Gibbs association based NHPP tracker in [146] with comparable tracking accuracy whilst consuming much less computational time. Compared to the PMHT method in [151], our method is an online Bayesian approximate filtering method and can offer the approximate posterior of target state and association variables. In comparison to the ET-JPDA filter in [147], our method systematically approximates the exact posterior with a mathematically defined objective function to minimise a certain Kullback-Leibler (KL) divergence; results demonstrate that our method has much more accurate tracking performance with less track loss and computational time.

Whilst the standard VB-AbNHPP tracker assumes a known static parameter, we also develop an adaptive tracker that can cope with unknown target and clutter rates. In particular, the independent Gamma initial priors are used to model targets and clutter Poisson rates, under which the tractable variational updates are kept within our framework. Subsequently, we develop Algorithm 9 that can simultaneously perform the tracking and Poisson rates learning tasks. Comprehensive derivations for the proposed tracker are included, illustrating a rational initialisation of the variational distribution and the development of an efficient-to-implement (yet demanding-to-derive) variational lower bound. Other parameters such as the measurement covariance can be learnt similarly if required.

Two other contributions of this chapter are that we construct an association-based NHPP system and provide a general coordinate ascent variational filtering framework with online static parameter learning. To start with, this chapter concisely states the relationship between the association-based NHPP measurement model and the original NHPP model in [144], and provides a dynamic Bayesian network formulation to facilitate the understanding of the dependence structure of the model that may provide further insights on the applicable approximate inference method. Moreover, in Section 5.3, we consider a general dynamic system and provide a unified framework for

coordinate ascent variational filtering with or without static parameter learning, whilst other similar variational filtering strategies in [51–53] are only for different or specific dynamic systems.

5.1.2 Chapter outline

The rest of chapter is organised as follows. Section 5.2 defines the association-based NHPP system and formulate the tracking problem. Section 5.3 introduces a general coordinate ascent variational filtering framework that allows approximate filtering with or without static parameter learning. Subsequently, we derive the VB-AbNHPP tracker with targets and clutter Poisson rates learning in Section 5.4. Following similar steps, in Section 5.5, we briefly derive the standard VB-AbNHPP tracker, which assumes known Poisson rates. Section 5.6 demonstrates the performance of both VB-AbNHPP tracker with and without the known rate using simulated data. Finally, Section 5.7 concludes the chapter.

5.2 Problem formulation

Assume that there are K targets. At time step n, their joint state is

$$X_n = [X_{n,1}^\top, X_{n,2}^\top, ..., X_{n,K}^\top]^\top,$$

where each vector $X_{n,k}, k \in \{1, ..., K\}$ denotes the kinematic state (e.g. position and velocity) for the k-th target. Let $Y_n = [Y_{n,1}, ..., Y_{n,M_n}]$ denote measurements received at time step n, and M_n is the total number of measurements.

This chapter is based on the NHPP measurement model proposed in [144]. Denote the set of Poisson rates by $\Lambda = [\Lambda_0, \Lambda_1, ..., \Lambda_K]$, where Λ_0 is the clutter rate and Λ_k is the k-th target rate, $k = 1, ..., K$. Each target k generates measurements by a NHPP with a Poisson rate Λ_k, and the total measurement process is also a NHPP from the superposition of the conditional independent NHPP measurement processes from K targets and clutter. The total number of measurements follows a Poisson distribution with rate $\Lambda_{\text{sum}} = \sum_{k=0}^{K} \Lambda_k$.

The likelihood function deduced in [144] is

$$h(Y_n, M_n | X_n, \Lambda) = \frac{e^{-\Lambda_{\text{sum}}}}{M_n!} \prod_{j=1}^{M_n} (\sum_{k=0}^{K} \Lambda_k \ell(Y_{n,j} | X_{n,k})). \tag{5.1}$$

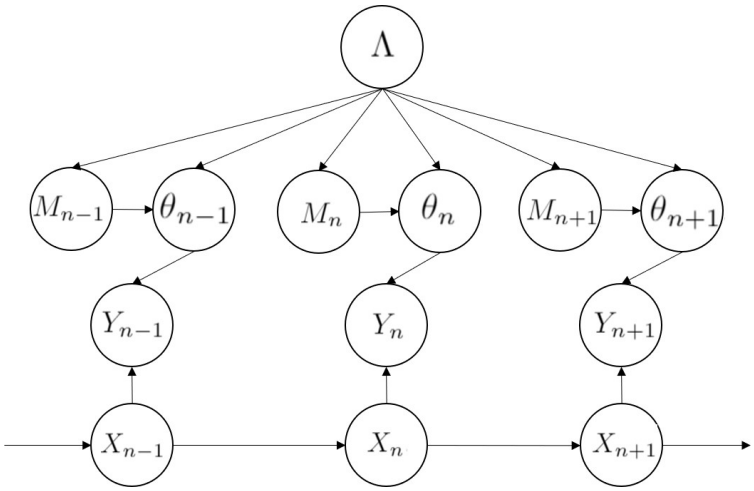


Fig. 5.1 Dynamic Bayesian network representing the joint distribution over associations θ_n, Poisson rates Λ, target states X_n, measurement number M_n, and measurement set Y_n

where we assume the target originated measurement follows a linear Gaussian model while clutters are uniformly distributed in the observation area V:

$$\ell(Y_{n,j}|X_{n,k}) = \begin{cases} \mathcal{N}(HX_{n,k}, R_k), & k \neq 0; \quad \text{(object)} \\ \frac{1}{V}, & k = 0; \quad \text{(clutter)} \end{cases} \tag{5.2}$$

where $X_{n,0}$ denotes the parameter/information of the clutter likelihood (e.g. in our case, the region of the uniform distribution), and in this chapter, we assume it is always known. Note that $X_{n,0}$ is not included in the joint target state X_n. For point target k, R_k indicates the sensor noise, and for extended target, R_k is the extended target's covariance/extent.

5.2.1 Association-based NHPP measurement model

Here we reformulate the NHPP measurement model by incorporating the association variables. First, we define the association variable $\theta_n = [\theta_{n,1}, ..., \theta_{n,M_n}]$, with each component $\theta_{n,j} \in \{0, 1, ..., K\}$; $\theta_{n,j} = 0$ indicates that $Y_{n,j}$ is generated by clutter, and $\theta_{n,j} = k, k \in \{1, \ldots, K\}$ means that $Y_{n,j}$ is generated from the target k. Clearly, θ_n and Y_n have the same length M_n. Subsequently, the joint distribution with the association variables θ_n is

$$p(Y_n, M_n, \theta_n|X_n, \Lambda) = p(Y_n|\theta_n, X_n)p(\theta_n|M_n, \Lambda)p(M_n|\Lambda), \tag{5.3}$$

where $p(M_n|\Lambda)$ is a Poisson distribution

$$p(M_n|\Lambda) = \frac{\exp(-\Lambda_{\text{sum}})(\Lambda_{\text{sum}})^{M_n}}{M_n!}, \tag{5.4}$$

and all these M_n measurements are conditionally independent

$$p(Y_n|\theta_n, X_n) = \prod_{j=1}^{M_n} \ell(Y_{n,j}|X_{n,\theta_{n,j}}), \tag{5.5}$$

where M_n is known from θ_n, and the function ℓ is the same as in (5.1) and defined in (5.2). For association θ_n we define

$$p(\theta_n|M_n, \Lambda) = \prod_{j=1}^{M_n} p(\theta_{n,j}|\Lambda), \tag{5.6}$$

where each association component is categorical distributed

$$p(\theta_{n,j}|\Lambda) = \frac{\sum_{k=0}^{K} \Lambda_k \delta[\theta_{n,j} = k]}{\Lambda_{\text{sum}}}. \tag{5.7}$$

We can see that this measurement model formulation is theoretically equivalent to the NHPP model in [144]. To demonstrate it, we use the definition from (5.3) to (5.7); by marginalising the association θ_n out from $p(Y_n, M_n, \theta_n|X_n, \Lambda)$, we can find out that it is equal to the likelihood function in (5.1):

$$\sum_{\theta_n} p(Y_n, M_n, \theta_n|X_n, \Lambda) = h(Y_n, M_n|X_n, \Lambda). \tag{5.8}$$

Therefore, these two formulations are equivalent.

5.2.2 Dynamic Bayesian network modelling

The association-based NHPP measurement model can be combined with any dynamic models introduced in chapters 2 and 3 for multi-target state X_n to formulate a complete dynamic Bayesian network. In this chapter, we consider a standard independent linear

Gaussian transition for each target state $X_{n,k}$, i.e.

$$\begin{aligned} p(X_n|X_{n-1}) &= \prod_{k=1}^{K} p(X_{n,k}|X_{n-1,k}) \\ &= \prod_{k=1}^{K} \mathcal{N}(X_{n,k}; F_{n,k}X_{n-1,k} + B_{n,k}, Q_{n,k}). \end{aligned} \tag{5.9}$$

Corresponding to the multi-target tracking problem under the association-based NHPP measurement model, the target distribution from time step 0 to N can be factorised as follows

$$\begin{aligned} p(X_{0:N}, Y_{1:N}, \theta_{1:N}, M_{1:N}|\Lambda) =& p(X_0) \prod_{n=1}^{N} p(X_n|X_{n-1}) \\ & \times p(Y_n|\theta_n, X_n)p(\theta_n|M_n, \Lambda)p(M_n|\Lambda) \end{aligned} \tag{5.10}$$

where $p(X_n|X_{n-1})$ is the transition density specified by the dynamic model (e.g. constant velocity), and the other distributions are given in (5.4)-(5.7).

This joint distribution can be represented by a dynamic Bayesian network (DBN) shown in Fig. 5.1, based on the factorisation in (5.10). Each conditional distribution on the right hand side of (5.10) can be depicted by the directed arrows, which point from the parent nodes to the child nodes (e.g. the arrow is from Λ to M_n for the conditional distribution $p(M_n|\Lambda)$).

Conditional on known parameters $K, R_{1:K}$ and the transition $p(X_n|X_{n-1})$, the objective of filtering is to sequentially estimate the posterior $p(X_n, \theta_n, \Lambda|Y_{1:n})$, which is equivalent to $p(X_n, \theta_n, \Lambda|Y_{1:n}, M_{1:n})$ conditional on $M_{1:n}$ since cardinality M_n is inherently known once the measurements Y_n are received. The exact filtering strategy for the association-based NHPP tracker can then be expressed as follows

$$\begin{aligned} p(X_n, \theta_n, \Lambda|Y_{1:n}) \propto& p(Y_n|\theta_n, X_n)p(\theta_n|M_n, \Lambda)p(M_n|\Lambda) \\ & \times \int p(X_{n-1}, \Lambda|Y_{1:n-1})p(X_n|X_{n-1})dX_{n-1}, \end{aligned} \tag{5.11}$$

where parameters $K, R_{1:K}$ are known and thus are implicitly conditioned and omitted from all relevant densities for convenience. The explicit evaluation of (5.11) is intractable since it requires enumerating all possible configurations of $\theta_{1:n}$. In this thesis, the VB-AbNHPP tracker approximates the online filtering recursion (5.11) with the coordinate ascent variational Bayes technique.

5.3 Coordinate ascent variational filtering with online parameter learning

In this section, we will introduce the coordinate ascent variational filtering framework in a general setting before we apply it to the tracking problem formulated in Section 5.2. We consider a dynamic system with several static parameters, a sequence of latent states, and measurements $\mathcal{Y}_{1:n}$ from time step 1 to n. We denote $\mathcal{Z}_n = \mathcal{X}_n \bigcup \Xi$ as the set of all latent variables in the system at time step n that we wish to infer, where $\mathcal{X}_n$ is the set of all unknown sequential latent states at time step n, and Ξ is the set of all unknown static parameter(s) of the system. We assume that elements in all defined sets are distinguishable.

Assume that $\Xi, \mathcal{X}_1, ..., \mathcal{X}_n$ are disjoint with each other. Note that $\mathcal{Z}_1, ..., \mathcal{Z}_n$ by our definition are not mutually disjoint unless all system static parameters are known; in this special case, $\Xi = \emptyset$ and thus $\mathcal{Z}_n = \mathcal{X}_n$. Finally, we assume that the exact optimal filtering with online parameter estimation for this system can be recursively expressed by the prediction step and the update step, which yield the following predictive prior $p_{n|n-1}(\mathcal{Z}_n)$ and posterior $p(\mathcal{Z}_n|\mathcal{Y}_{1:n})$, respectively, i.e.

$$\begin{aligned} p_{n|n-1}(\mathcal{Z}_n) &= \int f(\mathcal{X}_n|\mathcal{Z}_{n-1}) p(\mathcal{Z}_{n-1}|\mathcal{Y}_{1:n-1}) d\mathcal{X}_{n-1}, \\ p(\mathcal{Z}_n|\mathcal{Y}_{1:n}) &\propto g(\mathcal{Y}_n, \mathcal{Z}_n) p_{n|n-1}(\mathcal{Z}_n), \end{aligned} \tag{5.12}$$

where f is a known conditional probability density of $\mathcal{X}_n$ conditional on $\mathcal{Z}_{n-1}$. g is an arbitrary known function that depends on $\mathcal{Y}_n$ and $\mathcal{Z}_n$. It can represent an unnormalised likelihood function, and can even incorporate known functions that are related to $\mathcal{Z}_n$ instead of directly involving $\mathcal{Y}_n$. Note that f and g may also depend on other known parameters. p is defined as the exact probability law of the considered dynamic system; here p is the same probability law as defined in Section 5.2 in the case of a NHPP system described in Section 5.2.

A typical example of the considered system is a general state space model where the unknown latent state $\mathcal{X}_{1:n}$ follows a first-order Markovian transition and Ξ refers to the unknown parameter(s) of the transition and/or measurement function. Moreover, this system also applies to our AbNHPP framework formulated in Section 5.2. Specifically, $\mathcal{Y}_n$ refers to the measurements Y_n, $\mathcal{X}_n$ refers to the set $\{X_n, \theta_n\}$, and Ξ refers to the parameter set $\{\Lambda, R_{1:K}\}$ or simply $\emptyset$ depending on whether those parameters are required to be inferred.

Variational filtering can be employed when the exact filtering recursion in (5.12) is intractable, which recursively approximates $p(\mathcal{Z}_n|\mathcal{Y}_{1:n})$ in (5.12) with a converged variational distribution $q_n^*(\mathcal{Z}_n)$ that is chosen by minimising a certain KL divergence. By convention, we divide our variational filtering framework into a prediction step and an update step. In the following subsections, we will first describe the objective and rationale of our approximate filtering strategy in Section 5.3.1, then clarify the details of the variational update step in Section 5.3.2, and finally present the way to perform a reliable prediction step in Section 5.3.3. The framework's applicability will be discussed in Section 5.3.4.

5.3.1 Approximate filtering objective and probability law

Due to the unavailability of an exact $p(\mathcal{Z}_{n-1}|\mathcal{Y}_{1:n-1})$, the exact predictive prior $p_{n|n-1}(\mathcal{Z}_n)$ in (5.12) is intractable. Therefore, the prediction step in our approximate filtering aims to approximate the exact predictive prior $p_{n|n-1}(\mathcal{Z}_n)$ with a tractable distribution $\hat{p}_{n|n-1}(\mathcal{Z}_n)$, whose construction will be discussed in Section 5.3.3. With such a $\hat{p}_{n|n-1}(\mathcal{Z}_n)$, the approximate filtering probability law for the update step at the time step n can be defined as $\hat{p}_n$, which satisfies the following factorisation:

$$\hat{p}_n(\mathcal{Z}_n, \mathcal{Y}_n) \propto g(\mathcal{Y}_n, \mathcal{Z}_n)\hat{p}_{n|n-1}(\mathcal{Z}_n), \tag{5.13}$$

Specifically, $\hat{p}_n$ is defined for variables in $\mathcal{Y}_n$ and the set $\mathcal{Z}_n$, and g is the same function as in the exact filtering recursion in (5.12). Note that by such a construction, we have $\hat{p}_n(\mathcal{Z}_n) = \hat{p}_{n|n-1}(\mathcal{Z}_n)$ only if the function $\int g(\mathcal{Y}_n, \mathcal{Z}_n)d\mathcal{Y}_n$ does not depend on $\mathcal{Z}_n$, e.g. when $g(\mathcal{Y}_n, \mathcal{Z}_n)$ is a conditional density $\hat{p}_n(\mathcal{Y}_n|\mathcal{Z}_n)$. This factorisation in (5.13) also directly suggests the following posterior:

$$\hat{p}_n(\mathcal{Z}_n|\mathcal{Y}_n) \propto g(\mathcal{Y}_n, \mathcal{Z}_n)\hat{p}_{n|n-1}(\mathcal{Z}_n). \tag{5.14}$$

By comparing (5.14) with (5.12), it can be seen that the posterior under the approximate filtering probability law, i.e. $\hat{p}_n(\mathcal{Z}_n|\mathcal{Y}_n)$, is identical to the exact posterior $p(\mathcal{Z}_n|\mathcal{Y}_{1:n})$ if $\hat{p}_{n|n-1}(\mathcal{Z}_n)$ equals $p_{n|n-1}(\mathcal{Z}_n)$. Therefore, $\hat{p}_n(\mathcal{Z}_n|\mathcal{Y}_n)$ is regarded as the target distribution of the approximate filtering, since it is expected to provide a close approximation to the exact posterior $p(\mathcal{Z}_n|\mathcal{Y}_{1:n})$ in (5.12) if the prediction step produces an accurate $\hat{p}_{n|n-1}(\mathcal{Z}_n)$. Subsequently, the objective of the update step in our approximate filtering is to infer the target distribution $\hat{p}_n(\mathcal{Z}_n|\mathcal{Y}_n)$ via (5.14).

5.3.2 Update step with coordinate ascent variational inference

In the update step, a converged variational distribution $q_n^*(\mathcal{Z}_n)$ is evaluated to approximate the target distribution $\hat{p}_n(\mathcal{Z}_n|\mathcal{Y}_n)$ in (5.14) within the coordinate ascent variational inference framework. To this end, we first posit a (mean-field) family of variational distributions $q_n(\mathcal{Z}_n)$. Each member of this family must satisfy the following factorisation $q_n(\mathcal{Z}_n) = \prod_{i=1}^{\nu} q_n^i(z_n^i)$, where each z_n^i $(i = 1, 2, ..., \nu)$ is a predefined factorised variable and is disjointly partitioned from $\mathcal{Z}_n$, i.e. $\{z_n^1, z_n^2, ..., z_n^\nu\} = \mathcal{Z}_n$. It is noted that here we adopted a mean-field approximation, which is a special case of the more versatile conditionally factorised variational Bayes presented in Chapter 4, which can theoretically yield a better approximation with an adjustable conditional variational distribution, preserving the dependence between variables. Then our variational distribution $q_n^*(\mathcal{Z}_n)$ is chosen from the posited family that maximises the following evidence lower bound (ELBO) $\mathcal{F}_n(q_n)$:

$$\mathcal{F}_n(q_n) = \mathrm{E}_{q_n(\mathcal{Z}_n)} \log \frac{g(\mathcal{Y}_n, \mathcal{Z}_n)\hat{p}_{n|n-1}(\mathcal{Z}_n)}{q_n(\mathcal{Z}_n)}. \tag{5.15}$$

The rationale behind this optimisation is to minimise $\mathrm{KL}(q_n(\mathcal{Z}_n)||\hat{p}_n(\mathcal{Z}_n|\mathcal{Y}_n))$, since this KL divergence can be expressed as the ELBO $\mathcal{F}_n(q_n)$ in (5.15) plus a constant that, according to (5.14), is not dependent on $\mathcal{Z}_n$.

The optimisation of $\mathcal{F}_n(q_n)$ in (5.15) with respect to q_n can be done by the following coordinate ascent algorithm. We start by setting $q_n^i(z_n^i)$ to an initialised distribution $q_n^{(0)}(z_n^i)$ for all $i = 1, 2, ..., \nu$; then we iteratively update q_n^i for each $i = 1, 2, ..., \nu$ according to (5.16) while keeping $q_n^{i-}(z_n^{i-})$ fixed, where $q_n^{i-}(z_n^{i-})$ is defined as the joint variational distribution of all variables in $\mathcal{Z}_n$ except z_n^i, i.e. $q_n^{i-}(z_n^{i-}) = \prod_{j\neq i} q_n^j(z_n^j)$.

$$q_n^i(z_n^i) \propto \exp\left(\mathrm{E}_{q_n^{i-}(z_n^{i-})} \log g(\mathcal{Y}_n, \mathcal{Z}_n)\hat{p}_{n|n-1}(\mathcal{Z}_n)\right). \tag{5.16}$$

The q_n^i in (5.16) is the optimal distribution that achieves the highest ELBO $\mathcal{F}_n(q_n)$ when q_n^{i-} is fixed. Each update via (5.16) guarantees an increment of $\mathcal{F}_n(q_n)$ so that the algorithm eventually finds a local optimum. Such an optimisation procedure is known as CAVI. More details about CAVI, including the derivation of (5.16), can be found in [12, 23].

The convergence of the CAVI can be assessed by monitoring the ELBO $\mathcal{F}_n(q_n)$ in (5.15) for each iteration of updates. Typically, when the increment of $\mathcal{F}_n(q_n)$ is smaller than a certain threshold, we assume CAVI has converged and the latest updated $q_n(\mathcal{Z}_n)$ is chosen as our approximate filtering result $q_n^*(\mathcal{Z}_n)$.

5.3.3 Prediction step with approximate filtering prior

Recall that the objective of our prediction step is to find a $\hat{p}_{n|n-1}(\mathcal{Z}_n)$ that approximates the exact predictive prior $p_{n|n-1}(\mathcal{Z}_n)$ in (5.12), whose intractability arises from the lack of an exact filtering prior $p(\mathcal{Z}_{n-1}|\mathcal{Y}_{1:n-1})$. To obtain a tractable predictive prior, we replace this intractable $p(\mathcal{Z}_{n-1}|\mathcal{Y}_{1:n-1})$ in the integrand of (5.12) with some tractable approximate filtering prior $r(\mathcal{Z}_{n-1})$. A natural choice for such an approximate filtering prior is $q^*_{n-1}(\mathcal{Z}_{n-1})$, which is the converged variational distribution obtained from the approximate filtering at the time step t_{n-1}, and subsequently our approximate predictive prior can be written as (5.17). However, in some cases, a more sophisticated construction of the approximate filtering prior is required to ensure a reliable static parameter estimator. Below, we discuss this issue by considering two different scenarios depending on whether the online estimation of the static parameters is required.

5.3.3.1 $\Xi = \emptyset$

In this case, we assume that all static parameters in the system are known. Thus, we have $\mathcal{Z}_{n-1} = \mathcal{X}_{n-1}$, and our algorithm focuses solely on performing the approximate filtering task. To accomplish this, we suggest using the standard approximate filtering prior $r(\mathcal{Z}_{n-1}) = q^*_{n-1}(\mathcal{Z}_{n-1})$ to evaluating the $\hat{p}_n(\mathcal{Z}_n)$ for the prediction step:

$$\hat{p}_{n|n-1}(\mathcal{Z}_n) = \int f(\mathcal{X}_n|\mathcal{Z}_{n-1}) q^*_{n-1}(\mathcal{Z}_{n-1}) d\mathcal{X}_{n-1}. \tag{5.17}$$

In other words, the converged variational distribution for the last approximate filtering step is employed as the prior for our current prediction step. Such an empirical approximate filtering prior is commonly used in many approximate filtering methods such as the extended Kalman filter, and other Gaussian approximate filters [36].

5.3.3.2 $\Xi \neq \emptyset$

Our algorithm in this setting performs the approximate filtering along with online Bayesian parameter learning for the static parameters Ξ. Recall that examples of Ξ can be the Poisson rate and measurement covariance/extent $\{\Lambda, R_{1:K}\}$ in the tracking task in Section 5.2. Specifically, the parameter posterior $p(\Xi|\mathcal{Y}_{1:n})$ is approximated by $q^*_n(\Xi) = \int q^*_n(\mathcal{Z}_n) d\mathcal{X}_n$. It has been observed that applying the standard approximate filtering prior in (5.17) to this setting (i.e. set $\hat{p}_{n|n-1}(\mathcal{Z}_n) = \int f(\mathcal{X}_n|\mathcal{Z}_{n-1}) q^*_{n-1}(\Xi, \mathcal{X}_{n-1}) d\mathcal{X}_{n-1}$ for all time steps) may cause the variance of $q^*_n(\Xi)$ to be very small before its mean converges to the ground truth of Ξ; such an overconfident $q^*_n(\Xi)$ renders it very difficult

to correct our estimation of Ξ with future data. To mitigate this issue, it is empirically helpful to slightly 'flat' $q_n^*(\Xi)$ so that it can be less confident. Therefore, we construct our approximate filtering prior $r(\mathcal{Z}_{n-1})$ as follows,

$$r(\mathcal{Z}_{n-1}) \propto q_{n-1}^*(\Xi)^{\gamma_{n-1}} q_{n-1}^*(\mathcal{X}_{n-1}|\Xi), \tag{5.18}$$

where $q_{n-1}^*(\mathcal{X}_{n-1}|\Xi) = q_{n-1}^*(\mathcal{X}_{n-1}, \Xi)/q_{n-1}^*(\Xi)$, and henceforth the approximate predictive prior $\hat{p}_n(\mathcal{Z}_n)$ is

$$\begin{aligned} \hat{p}_{n|n-1}(\mathcal{Z}_n) &= \int f(\mathcal{X}_n|\mathcal{Z}_{n-1}) r(\mathcal{Z}_{n-1}) d\mathcal{X}_{n-1} \\ &\propto q_{n-1}^*(\Xi)^{\gamma_{n-1}} \int f(\mathcal{X}_n|\Xi, \mathcal{X}_{n-1}) q_{n-1}^*(\mathcal{X}_{n-1}|\Xi) d\mathcal{X}_{n-1}. \end{aligned} \tag{5.19}$$

$\gamma_{n-1} \in (0, 1]$ is a forgetting factor that can 'smooth' the density $q_{n-1}^*(\Xi)$ when used in the approximate filtering prior $r(\mathcal{Z}_{n-1})$. Specifically, $\gamma_{n-1} = 1$ recovers the standard approximate filtering prior $q_{n-1}^*(\mathcal{Z}_{n-1})$ as used in (5.17), and in the limiting case $\gamma_{n-1} = 0$, the approximate filtering prior $r(\mathcal{Z}_{n-1})$ completely ignores the $q_{n-1}^*(\Xi)$ and assumes a uniformly distributed Ξ. Moreover, it is useful to construct a monotonically increasing sequence $\{\gamma\}_n$ that approaches 1 when n is large, such that our parameter posterior approximation $q_n^*(\Xi)$ would eventually converge to the exact value of Ξ as a point estimator.

5.3.4 Further discussions on the applicability

The introduced coordinate ascent variational filtering with parameter learning framework is devised for a general dynamic system that satisfies the optimal filtering recursion in (5.12). However, not all such systems can lead to closed-form approximated distributions through the introduced framework. The key to yielding a tractable approximate distribution is to make sure that the choice of the factorised variable $z_n^i (i = 1, 2, ..., \nu)$ and the $\hat{p}_{n|n-1}(\mathcal{Z}_n)$ produced by our prediction step should lead to an analytically tractable updated variational distribution q_n^i in (5.16). This may require the predictive prior $\hat{p}_{n|n-1}(\mathcal{Z}_n)$ to take some particular parametric form that does not agree with the two constructions of $\hat{p}_n(\mathcal{Z}_n)$ we suggested in Section 5.3.3 (i.e. (5.17) and (5.19)). In this case, we could adopt a $\hat{p}_{n|n-1}(\mathcal{Z}_n)$ in the desired parametric form, but ensure that it matches the moments of the predictive distribution suggested in Section 5.3.3 to preserve some accuracy. Alternatively, any intractable coordinate ascent update in (5.16) may be approximated by the Monte Carlo methods. In particular, we can sample from the intractable variational distributions to approximate other updated

variational distributions in a closed-form expression. Such a strategy is commonly seen in the Monte Carlo based CAVI, e.g. [133, 127, 53, 59].

5.4 Variational Bayes AbNHPP tracker

The approximate inference for the association based NHPP system in Section 5.2 can now be carried out within the coordinate ascent variational filtering framework in Section 5.3. In particular, our VB-AbNHPP tracker with known static system parameters (i.e. $\Xi = \emptyset$) will be presented in Section 5.5 for tracking tasks with known Poisson rate Λ and measurement covariance/extent $R_{1:K}$. In this section, we demonstrate that our VB-AbNHPP tracker can perform tracking and online learning the unknown parameters within the variational filtering framework presented in Section 5.3. We will consider the tracking tasks with an unknown Poisson rate Λ. The measurement covariance/extent $R_{1:K}$ can be learnt in a similar fashion if unknown, but for now we assume it is known for simplicity.

From now on, we assume target number K and measurement covariance $R_{1:K}$ in the association based NHPP system in Section 5.2 are all known, and unless otherwise stated, these parameters are always implicitly conditioned. Subsequently, the variables of the general dynamic system in Section 5.3 are now $\Xi = \{\Lambda\}$, $\mathcal{Y}_n = Y_n$, $\mathcal{X}_n = \{X_n, \theta_n\}$ and $\mathcal{Z}_n = \{X_n, \theta_n, \Lambda\}$. Comparing the optimal filter recursion (5.12) to the exact optimal filter recursion for the association based NHPP system in (5.11), the densities for f, g become

$$\begin{aligned} f(\mathcal{X}_n|\mathcal{Z}_{n-1}) =& p(\theta_n|M_n, \Lambda)p(X_n|X_{n-1}), \\ g(\mathcal{Y}_n, \mathcal{Z}_n) =& p(Y_n|\theta_n, X_n)p(M_n|\Lambda). \end{aligned} \tag{5.20}$$

Note that f and g are known when the approximate filtering is carried out at time step n, where M_n is inherently obtained from the latest received observations Y_n. Since $\Xi \neq \emptyset$, we construct our approximate predictive prior $\hat{p}_{n|n-1}(\mathcal{Z}_n)$ according to (5.19) in Section 5.3.3.2, i.e.

$$\hat{p}_{n|n-1}(X_n, \theta_n, \Lambda) \propto q^*_{n-1}(\Lambda)^{\gamma_{n-1}} p(\theta_n|M_n, \Lambda) \int p(X_n|X_{n-1}) q^*_{n-1}(X_{n-1}|\Lambda) dX_{n-1}, \tag{5.21}$$

where the forgetting factor $\gamma_{n-1} \in (0, 1]$ was introduced in Section 5.3.3.2. Now we assume the factorisation for our mean-field family is $q(X_n, \theta_n, \Lambda) = q(X_n)q(\theta_n)q(\Lambda)$. This factorisation will later result in tractable coordinate ascent updates, as we will see later. It also suggests that $q^*_{n-1}(X_{n-1}|\Lambda) = q^*_{n-1}(X_{n-1})$. Subsequently, the approximate

predictive prior in (5.21) can be rewritten in a factorised form as follows for the convenience of later derivation,

$$\begin{aligned}\hat{p}_{n|n-1}(X_n,\theta_n,\Lambda)=&\hat{p}_{n|n-1}(X_n)\hat{p}_{n|n-1}(\Lambda)\hat{p}_{n|n-1}(\theta_n|\Lambda),\\ \hat{p}_{n|n-1}(X_n)=&\int p(X_n|X_{n-1})q^*_{n-1}(X_{n-1})dX_{n-1},\\ \hat{p}_{n|n-1}(\theta_n|\Lambda)=&p(\theta_n|M_n,\Lambda),\\ \hat{p}_{n|n-1}(\Lambda)\propto&q^*_{n-1}(\Lambda)^{\gamma_{n-1}}.\end{aligned} \tag{5.22}$$

According to Section 5.3.1 and (5.13), the approximate filtering law $\hat{p}_n$ of the update step for $Y_n, X_n, \theta_n, \Lambda$ is defined by,

$$\hat{p}_n(X_n,\theta_n,\Lambda,Y_n)\propto p(Y_n|\theta_n,X_n)p(M_n|\Lambda)p(\theta_n|M_n,\Lambda)\hat{p}_{n|n-1}(X_n)\hat{p}_{n|n-1}(\Lambda). \tag{5.23}$$

The prediction step described in Section 5.3.3 is now used to evaluate the approximate filtering priors in (5.22). Since $\hat{p}_{n|n-1}(\theta_n|\Lambda)$ in (5.22) can be directly obtained from our model in (5.6), the prediction step only requires evaluating $\hat{p}_{n|n-1}(X_n)$ and $\hat{p}_{n|n-1}(\Lambda)$ in (5.22). Their explicit forms will be given in (5.28) and (5.31) after a discussion of the conjugate prior.

5.4.1 Coordinate ascent update

Recall that our mean-field family satisfies $q(X_n,\theta_n,\Lambda)=q(X_n)q(\theta_n)q(\Lambda)$. Based on Section 5.3.2, the update step of our tracker aims to minimise the KL divergence $\mathrm{KL}(q_n(X_n)q_n(\theta_n)q(\Lambda)||\hat{p}_n(X_n,\theta_n,\Lambda|Y_n))$, or equivalently, to maximise the ELBO in (5.15) as follows

$$\begin{aligned}\mathcal{F}(q_n)=&\mathrm{E}_{q_n(X_n)q_n(\theta_n)q_n(\Lambda)}\log\frac{\hat{p}_{n|n-1}(X_n,\theta_n,\Lambda)}{q_n(X_n)q_n(\theta_n)q_n(\Lambda)}\\ &+\mathrm{E}_{q_n(X_n)q_n(\theta_n)q_n(\Lambda)}\log p(Y_n|\theta_n,X_n)p(M_n|\Lambda),\end{aligned} \tag{5.24}$$

where $\hat{p}_{n|n-1}(X_n,\theta_n,\Lambda)$ is specified in (5.22). To this end, it iteratively updates $q_n(X_n)$, $q_n(\Lambda)$ and $q_n(\theta_n)$ according to (5.16) until convergence. Despite not being an exact solution, it will be seen that all the updates are tractable and have closed-form solutions, which showcases the advantage of using coordinate ascent variational inference in this tracking task. We now derive these updates.

5.4.1.1 Update for $q_n(X_n)$

According to (5.16) on page 148, (5.5) on page 144 and (5.2) on page 143, and the fact that $q_n(\theta_n) = \prod_{j=1}^{M_n} q_n(\theta_{n,j})$ (this will later be shown in (5.33)), it yields the following update for X_n

$$\begin{aligned} q_n(X_n) \propto & \hat{p}_{n|n-1}(X_n) \exp\left(\mathrm{E}_{q_n(\theta_n)} \log p(Y_n|\theta_n, X_n)\right) \\ \propto & \hat{p}_{n|n-1}(X_n) \prod_{k=1}^{K} \mathcal{N}\left(\overline{Y}_n^k; HX_{n,k}, \overline{R}_n^k\right), \end{aligned} \tag{5.25}$$

where

$$\overline{R}_n^k = \frac{R_k}{\sum_{j=1}^{M_n} q_n(\theta_{n,j} = k)}, \tag{5.26}$$

$$\overline{Y}_n^k = \frac{\sum_{j=1}^{M_n} Y_{n,j} q_n(\theta_{n,j} = k)}{\sum_{j=1}^{M_n} q_n(\theta_{n,j} = k)}. \tag{5.27}$$

Such an update can be considered as updating the (predictive) prior $\hat{p}_{n|n-1}(X_n)$ in (5.22) with K pseudo-measurements $\overline{Y}_n^k, k = 1, 2, ..., K$ (defined in (5.27)), each generated independently from each target with a measurement covariance of (5.26). The conjugate prior for such an update is Gaussian. In fact, with the independent linear Gaussian transition $p(X_n|X_{n-1})$ in (5.9) and an independent initial Gaussian prior $p(X_0) = \prod_{k=1}^{K} p(X_{0,k})$, the updated variational distribution can always maintain an independent Gaussian form for each target (i.e. $q_n(X_n) = \prod_{k=1}^{K} q_n(X_{n,k})$).

Specifically, if we denote the converged variational distribution for the k-th target at time step $n-1$ as $q_{n-1}^*(X_{n-1,k}) = \mathcal{N}(X_{n-1,k}; \mu_{n-1|n-1}^{k*}, \Sigma_{n-1|n-1}^{k*})$, then its predictive prior is

$$\begin{aligned} \hat{p}_{n|n-1}(X_{n,k}) = & \mathcal{N}(X_{n,k}; \mu_{n|n-1}^{k*}, \Sigma_{n|n-1}^{k*}), \\ \mu_{n|n-1}^{k*} = & F_{n,k} \mu_{n-1|n-1}^{k*} + B_{n,k}, \\ \Sigma_{n|n-1}^{k*} = & F_{n,k} \Sigma_{n-1|n-1}^{k*} F_{n,k}^{\top} + Q_{n,k}, \end{aligned} \tag{5.28}$$

where $F_{n,k}, B_{n,k}, Q_{n,k}$ are given in (5.9). The variational distribution $q_n(X_{n,k})$ can now be updated by the Kalman filter:

$$\begin{aligned} q_n(X_{n,k}) &= \mathcal{N}(X_{n,k}; \mu^k_{n|n}, \Sigma^k_{n|n}), \\ T^k_n &= \overline{Y}^k_n - H\mu^{k*}_{n|n-1}, \\ S^k_n &= H\Sigma^{k*}_{n|n-1}H^\top + \overline{R}^k_n, \\ G &= \Sigma^{k*}_{n|n-1}H^\top S^{k^{-1}}_n, \\ \mu^k_{n|n} &= \mu^{k*}_{n|n-1} + GT^k_n, \\ \Sigma^k_{n|n} &= (I - GH)\Sigma^{k*}_{n|n-1}. \end{aligned} \tag{5.29}$$

Such an update can be independently carried out for all targets.

5.4.1.2 Update for $q_n(\Lambda)$

According to (5.16), (5.4) and (5.6), the variational distribution $q_n(\Lambda)$ is updated as follows,

$$\begin{aligned} q_n(\Lambda) \propto& \hat{p}_{n|n-1}(\Lambda) p(M_n|\Lambda) \exp\left(\mathrm{E}_{q_n(\theta_n)} \log p(\theta_n|M_n, \Lambda)\right) \\ =& \hat{p}_{n|n-1}(\Lambda) \frac{\exp(-\Lambda_{\text{sum}})(\Lambda_{\text{sum}})^{M_n}}{M_n!} \times \frac{1}{(\Lambda_{\text{sum}})^{M_n}} \\ & \times \exp\left(\mathrm{E}_{q_n(\theta_n)} \sum_{j=1}^{M_n} \sum_{k=0}^{K} \delta[\theta_{n,j} = k] \log \Lambda_k\right) \\ \propto& \hat{p}_{n|n-1}(\Lambda) \prod_{k=0}^{K} \exp(-\Lambda_k) \Lambda_k^{\sum_{j=1}^{M_n} q_n(\theta_{n,j}=k)}. \end{aligned} \tag{5.30}$$

Independent gamma turns out to be the conjugate prior for the likelihood function in (5.30). Specifically, if we assume an independent initial prior $p(\Lambda) = \prod_{k=0}^{K} p(\Lambda_k)$ where each $p(\Lambda_k)$ is a gamma distribution, then with prediction in (5.22) and update in (5.30), $q_n(\Lambda)$ is always in the independent Gamma form, i.e. $q_n(\Lambda) = \prod_{k=0}^{K} q_n(\Lambda_k)$ where each $q_n(\Lambda_k)$ is a gamma distribution. Denote $q^*_{n-1}(\Lambda_k) = \mathcal{G}(\Lambda_k; \eta^{k*}_{n-1|n-1}, \rho^{k*}_{n-1|n-1})$, where $\mathcal{G}(\eta, \rho)$ is the Gamma distribution with shape parameter η and scale parameter ρ. According to $\hat{p}_{n|n-1}(\Lambda)$ in (5.22), the predictive prior for each Λ_k is

$$\begin{aligned} \hat{p}_{n|n-1}(\Lambda_k) =& \mathcal{G}(\Lambda_k; \eta^{k*}_{n|n-1}, \rho^{k*}_{n|n-1}), \\ \eta^{k*}_{n|n-1} =& \eta^{k*}_{n-1|n-1}\gamma_{n-1} - \gamma_{n-1} + 1, \\ \rho^{k*}_{n|n-1} =& \rho^{k*}_{n-1|n-1}/\gamma_{n-1}, \end{aligned} \tag{5.31}$$

then the update for $q_n(\Lambda_k)$ is

$$\begin{aligned} q_n(\Lambda_k) =& \mathcal{G}(\Lambda_k; \eta^k_{n|n}, \rho^k_{n|n}), \\ \eta^k_{n|n} =& \eta^{k*}_{n|n-1} + \sum_{j=1}^{M_n} q_n(\theta_{n,j} = k), \\ \rho^k_{n|n} =& \rho^{k*}_{n|n-1}/(\rho^{k*}_{n|n-1} + 1). \end{aligned} \tag{5.32}$$

Consequently, each $q_n(\Lambda_k)$ can be updated independently.

5.4.1.3 Update for $q_n(\theta_n)$

The variational distribution $q_n(\theta_n)$ can be updated according to (5.16), (5.5) and (5.6), i.e.

$$\begin{aligned} q_n(\theta_n) &\propto \exp\left(\mathrm{E}_{q_n(\Lambda)q_n(X_n)} \log p(\theta_n|M_n, \Lambda) \log p(Y_n|\theta_n, X_n)\right) \\ &= \prod_{j=1}^{M_n} \exp\left(\mathrm{E}_{q_n(\Lambda)q_n(X_n)} \log p(\theta_{n,j}|\Lambda) \ell(Y_{n,j}|X_{n,\theta_{n,j}})\right) \\ &\propto \prod_{j=1}^{M_n} q_n(\theta_{n,j}), \end{aligned} \tag{5.33}$$

with each $q_n(\theta_{n,j})$ being

$$\begin{aligned} q_n(\theta_{n,j}) \propto& \exp\left(\mathrm{E}_{q_n(\Lambda)q_n(X_n)} \log p(\theta_{n,j}|\Lambda) \ell(Y_{n,j}|X_{n,\theta_{n,j}})\right) \\ \propto& \frac{\overline{\Lambda}_0}{V} \delta[\theta_{n,j} = 0] + \sum_{k=1}^{K} \overline{\Lambda}_k l_k \delta[\theta_{n,j} = k], \\ \overline{\Lambda}_k =& \exp(\mathrm{E}_{q_n(\Lambda)} \log \Lambda_k) = \exp(\psi(\eta^k_{n|n})) \rho^k_{n|n}, \\ l_k =& \mathcal{N}(Y_{n,j}; H\mu^k_{n|n}, R_k) \exp(-0.5 \mathrm{Tr}(R_k^{-1} H \Sigma^k_{n|n} H^\top)), \end{aligned} \tag{5.34}$$

where $\overline{\Lambda}_k$ is for $k = 0, 1, ..., K$, l_k is for $k = 1, ..., K$, and $\psi(\cdot)$ is the digamma function. The $\mu^k_{n|n}, \Sigma^k_{n|n}$ are given in (5.29), and (5.34) is obtained by substituting (5.7), (5.2), (5.29) and (5.32). It can now be seen that the variational distributions for each association variable are independent, and each of them is a categorical distribution specified (with a proportional constant) in (5.34). Similar to the updates for X_n, Λ, the updates for the association θ_n can also be independently carried out for each $\theta_{n,j}$.

5.4.2 Initialisation

As a deterministic algorithm, the local optimum (i.e. $q_n^*(X_n, \theta_n, \Lambda)$) found by the CAVI is sensitive to the initial variational distribution. A good initialisation can prevent the algorithm from being trapped in a bad local minimum and may also lead to faster convergence. Therefore, a good initial variational distribution is important for our tracker to perform an accurate and efficient multi-target tracking task.

A simple choice of initialisation is to employ the predictive distribution of target state $\hat{p}_{n|n-1}(X_n)$ to initialise $q_n(X_n)$. An alternative option utilises the prior distribution $p(\theta_n|M_n, \Lambda)$ in (5.6) to initialise $q_n(\theta_n)$. However, in our experiments, these straightforward initialisations often caused the CAVI to converge to undesirable local minima, resulting in track loss.

Therefore, this section presents an enhanced initialisation strategy for $q_n(\theta_n)$. Specifically, the setup for the initial association distribution $q_n^{(0)}(\theta_n)$ is detailed. After this initialisation, CAVI is performed by iteratively updating $q(\Lambda)$ and $q_n(X_n)$, which require the initial association distribution $q_n^{(0)}(\theta_n)$, introduced below, for the calculation of the update form.

The initial association distribution $q_n^{(0)}(\theta_n)$ is defined as follows. First, independent initial variational distributions are assumed assumes for each $\theta_{n,j}$ so as to be consistent with the updated form in (5.33), that is,

$$q_n^{(0)}(\theta_n) = \prod_{j=1}^{M_n} q_n^{(0)}(\theta_{n,j}) \tag{5.35}$$

Recall that the objective of the CAVI in our setup is to minimise the KL divergence between $q_n(X_n)q_n(\theta_n)q(\Lambda)$ and the target distribution $\hat{p}_n(X_n, \theta_n, \Lambda|Y_n)$. Therefore, the best (but intractable) initial distribution for $q_n(\theta_{n,j})$ may be the marginal target distribution $\hat{p}_n(\theta_{n,j}|Y_n)$. A suboptimal choice would be $\hat{p}_n(\theta_{n,j}|Y_{n,j})$, which, under the same approximate filtering law $\hat{p}_n$, only incorporates the information of the corresponding measurement $Y_{n,j}$ rather than all measurements Y_n. However, this suboptimal distribution is still intractable except for the special case that the Gamma prior $\hat{p}_n(\Lambda_k)$ shares the same scale parameter $\rho_{n|n-1}^{k*}$ in (5.31) for all $k = 1, 2, ..., K$. As a result,the intractable $\hat{p}_n(\theta_{n,j}|Y_{n,j})$ is further approximated with $\hat{p}_n(\theta_{n,j}|Y_{n,j}, \Lambda = \hat{\Lambda})$, which can be regarded as an empirical Bayes approximation. Specifically, the hyperparameter Λ (in $\theta_{n,j}$'s prior), instead of being marginalised out, is specified by the point estimate $\hat{\Lambda}$ that is obtained at the last time step $n-1$. It is expected that our initial distribution

$\hat{p}_n(\theta_{n,j}|Y_{n,j}, \Lambda = \hat{\Lambda})$ will be more accurate as n increases, since this point estimate $\hat{\Lambda}$ should gradually converge to the ground truth of Λ.

Therefore, by using $\hat{p}_n(\theta_{n,j}|Y_{n,j}, \Lambda = \hat{\Lambda})$ as empirical Bayes approximation to the suboptimal initial distribution for $q_n(\theta_{n,j})$, the designed initialisation strategy can be written as follows:

$$\begin{aligned}
q_n^{(0)}(\theta_{n,j}) =& \hat{p}_n(\theta_{n,j}|Y_{n,j}, \Lambda = \hat{\Lambda}) \propto \hat{p}_n(\theta_{n,j}, Y_{n,j}|\Lambda = \hat{\Lambda}) \\
=& p(\theta_{n,j}|\Lambda = \hat{\Lambda}) \int \ell(Y_{n,j}|X_{n,\theta_{n,j}}) \hat{p}_{n|n-1}(X_n) dX_n \\
=& \frac{\hat{\Lambda}_0}{V} \delta[\theta_{n,j} = 0] + \sum_{k=1}^{K} \hat{\Lambda}_k l_k^0 \delta[\theta_{n,j} = k], \qquad (5.36) \\
l_k^0 =& \mathcal{N}(Y_{n,j}; H\mu_{n|n-1}^{k*}, H\Sigma_{n|n-1}^{k*} H^\top + R_k), \\
\hat{\Lambda} =& \mathrm{E}_{q_{n-1}^*(\Lambda)} \Lambda = [\hat{\Lambda}_0, \hat{\Lambda}_1, ..., \hat{\Lambda}_K], \\
\hat{\Lambda}_k =& \mathrm{E}_{q_{n-1}^*(\Lambda)} \Lambda_k = \eta_{n-1|n-1}^{k*} \rho_{n-1|n-1}^{k*},
\end{aligned}$$

where l_k^0 is for $k = 1, 2, ..., K$, and $\hat{\Lambda}_k$ $(k = 0, 1, ..., K)$ is the estimate of Λ_k at the time step $n-1$. $\hat{p}_{n|n-1}(X_n)$ is the predictive prior and $\mu_{n|n-1}^{k*}, \Sigma_{n|n-1}^{k*}$ are given in (5.28). Each $q_n^{(0)}(\theta_{n,j})$ is set to be $\hat{p}_n(\theta_{n,j}|Y_{n,j}, \Lambda = \hat{\Lambda})$, where $\hat{p}_n$ is the approximate filtering probability law defined in Section 5.3.1 and (5.23). The evaluation of $\hat{p}_n(\theta_{n,j}|Y_{n,j}, \Lambda = \hat{\Lambda})$ in (5.36) is a direct result of (5.23), and its detailed derivation is shown in Appendix 5.A. This initial variational distribution $q_n^{(0)}(\theta_{n,j})$ has a similar form as the updated $q_n(\theta_n)$ in (5.33) and (5.34), and can also be independently evaluated for each $\theta_{n,j}$.

5.4.3 ELBO computation

It is useful to compute the ELBO in (5.24) for each recursion of updates. The reasons are: 1) It provides a convenient way to monitor the convergence. 2) It is practically useful to check the implementation of the algorithm. 3) It reflects the quality of the found variational distribution (for another example of this benefit of calculating ELBO, see Chapter 6, where this property will be further illustrated and applied in Section 6.2 and 6.3). Recall that the ELBO is guaranteed to increase at each recursion of updates, and the objective of our tracker is to maximise the ELBO. By monitoring the increment of ELBO, we can assume that the algorithm converges once the increment of ELBO at the latest recursion falls below a certain threshold. Moreover, if the increment of ELBO resulting from any update is negative, the implementation of the algorithm must be incorrect.

The straightforward computation of the ELBO in (5.24) involves inversions of non-diagonal matrices $\Sigma^k_{n|n}$ and $\Sigma^{k*}_{n|n-1}$, which can be computationally prohibitive for a large number of targets. Here we adopt a relatively efficient way to evaluate the ELBO based on a more easy-to-compute marginal likelihood. Such a technique was first used to compute the ELBO in Section 5.3.7 in [128]. Specifically, if the $q(X_n)$ is the latest updated variational distribution (i.e. $q(\theta_n)$ has not been updated with the current $q(X_n)$), the ELBO in (5.24) can be exactly expressed as follows,

$$\begin{aligned}\mathcal{F}(q_n) =& \sum_{j=1}^{M_n}\sum_{k=0}^{K} q(\theta_{n,j}=k)\left(\psi(\eta^k_{n|n}) + \log\frac{\rho^k_{n|n}}{q(\theta_{n,j}=k)}\right)\\ &- \frac{1}{2}\sum_{j=1}^{M_n}\sum_{k=1}^{K} q_n(\theta_{n,j}=k)\left(Y_{n,j}^\top R_k^{-1} Y_{n,j} + \log\det R_k\right)\\ &+ \frac{1}{2}\sum_{k=1}^{K}\left(\overline{Y}_n^{k\top}\overline{R}_n^{k-1}\overline{Y}_n^k - T_n^{k\top}S_n^{k-1}T_n^k + \log\frac{\det\overline{R}_n^k}{\det S_n^k}\right)\\ &+ \left[\frac{D}{2}\log 2\pi + \log\frac{1}{V}\right]\sum_{j=1}^{M_n} q_n(\theta_{n,j}=0)\\ &- \sum_{k=0}^{K}\eta^k_{n|n}\rho^k_{n|n} - \mathrm{KL}(q_n(\Lambda)||\hat{p}_{n|n-1}(\Lambda))\\ &- \frac{1}{2}DM_n\log 2\pi - \log(M_n!),\end{aligned} \tag{5.37}$$

$$\begin{aligned}&\mathrm{KL}(q_n(\Lambda)||\hat{p}_{n|n-1}(\Lambda))\\ =&\sum_{k=0}^{K}\Big[-\eta^{k*}_{n|n-1}\log\rho^k_{n|n} - \log\Gamma(\eta^k_{n|n}) + (\eta^k_{n|n}-\eta^{k*}_{n|n-1})\psi(\eta^k_{n|n}) + \eta^k_{n|n}(\rho^k_{n|n}/\rho^{k*}_{n|n-1}-1)\Big]\\ &+\sum_{k=0}^{K}\left(\log\Gamma(\eta^{k*}_{n|n-1}) + \eta^{k*}_{n|n-1}\log\rho^{k*}_{n|n-1}\right),\end{aligned} \tag{5.38}$$

where $\overline{Y}^k_n, \overline{R}^k_n$ are given in (5.26); (5.27), and T^k_n, S^k_n are given in the Kalman filter update (5.29), and $\eta^k_{n|n}, \rho^k_{n|n}, \eta^{k*}_{n|n-1}, \rho^{k*}_{n|n-1}$ are given in (5.30)(5.31). Note that all these terms have already been computed for updating the latest $q_n(X_n)$ and $q_n(\lambda)$. $\psi(\cdot)$ is the digamma function and $\Gamma(\cdot)$ is the gamma function. D is the dimension of the vector of a single measurement $Y_{n,j}$, and V, R_k are defined in (5.2). The laborious derivation for (5.37) is presented in Appendix 5.B. We emphasise again that the evaluation in (5.37) only equals the exact ELBO in (5.24) when $q_n(X_n)$ is the latest updated variational distribution.

Algorithm 9: VB-AbNHPP tracker with rate estimation at time step n

Require: $q^*_{n-1}(X_{n-1}), q^*_{n-1}(\Lambda), Y_n, M_n$, maximum iteration limit I, tolerance threshold $\epsilon > 0$.
Output: $q^*_n(X_n), q^*_n(\theta_n), q^*_n(\Lambda)$.
Prediction for X_n: Evaluate $\mu^{k*}_{n|n-1}, \Sigma^{k*}_{n|n-1}$ for $\hat{p}_{n|n-1}(X_n)$ via (5.28).
Prediction for Λ: Evaluate $\eta^{k*}_{n|n-1}, \rho^{k*}_{n|n-1}$ for $\hat{p}_{n|n-1}(\Lambda)$ via (5.31).
for $j = 1, 2, ..., M_n$ **do**
    Evaluate $q^{(0)}_n(\theta_{n,j})$ via (5.36), and initialise $q_n(\theta_{n,j}) \leftarrow q^{(0)}_n(\theta_{n,j})$.
end
for $i = 1, 2, ..., I$ **do**
    for $k = 1, 2, ..., K$ **do**
        Evaluate $\eta^k_{n|n}, \rho^k_{n|n}$ according to (5.32) for updating $q(\Lambda)$.
    end
    for $k = 1, 2, ..., K$ **do**
        Evaluate $\overline{R}^k_n, \overline{Y}^k_n$ according to (5.26)(5.27).
        Compute $\mu^k_{n|n}, \Sigma^k_{n|n}, T^k_n, S^k_n$ via (5.29) for updating $q_n(X_{n,k})$.
    end
    Evaluate the ELBO $\mathcal{F}^{(i)}_n$ according to (5.37).
    if $\mathcal{F}^{(i)}_n - \mathcal{F}^{(i-1)}_n < \epsilon \wedge i \geq 2$ **then**
        break
    end
    for $j = 1, 2, ..., M_n$ **do**
        Update $q_n(\theta_{n,j})$ according to (5.34).
    end
end
Set $q^*_n(X_n) = \prod_{k=1}^K q_n(X_{n,k})$, $q^*_n(\Lambda) = \prod_{k=1}^K q_n(\Lambda_k)$ and $q^*_n(\theta_n) = \prod_{j=1}^{M_n} q_n(\theta_{n,j})$.

The last lines in (5.37) and (5.38) are constants through each iteration of CAVI, and hence can be omitted when monitoring the increments of $\mathcal{F}(q_n)$. In a common tracking scenario where the dynamic transition in (5.9) and the positional measurement function are both independent for different coordinates, $R_k, \overline{R}^k_n, S^k_n$ are all diagonal matrices. Subsequently, all matrix inversions in (5.37) can be computed easily.

5.4.4 Algorithm

Finally, the approximate filtering algorithm at time step n of our VB-AbNHPP tracker is summarised in Algorithm 9. In brief, this algorithm first 1) evaluates the predictive distribution for X_n, Λ; then 2) sets the initialisation variational distribution for θ_n; and 3) carries out the CAVI until convergence. There are many steps in the algorithm that

can be parallelised. Specifically, the evaluation of $q_n^{(0)}(\theta_n)$ and the update of $q_n(\theta_n)$ can be paralleled for each association, and the update for $q_n(X_n)$ can be paralleled for each target. Moreover, the ELBO computation can also be further accelerated by exploring the parallel computing of the summands in (5.37).

The ELBO evaluation step (5.37) in Algorithm 9 can be neglected if the computational power is limited. In this case, we can either monitor the change in statistics of the variational distribution (e.g. the mean of $q(X_n)$) to check for convergence, or we can directly set the algorithm to perform a predefined number of iterations.

5.5 VB-AbNHPP tracker with known Poisson rates

By assuming known Poisson rates, the VB-AbNHPP tracker can be implemented with a simplified procedure. Since the derivation of this tracker is very similar to the that in Section 5.4, here we only present a brief derivation.

When the Poisson rate Λ is known, e.g. from prior knowledge or estimation results as in Section 5.4, the exact filtering strategy in (5.11) simplifies to

$$\begin{aligned} p(X_n, \theta_n | Y_{1:n}, \Lambda) \propto & p(Y_n|\theta_n, X_n) p(\theta_n|M_n, \Lambda) \\ & \times \int p(X_{n-1}|Y_{1:n-1}, \Lambda) p(X_n|X_{n-1}) dX_{n-1}. \end{aligned} \tag{5.39}$$

In this Section, similar to Section 5.4, we assume that the parameters K and $R_{1:K}$ are known and are always treated as implicit conditions throughout the equations and discussion. Subsequently, the variables defined for the general dynamic system considered in Section 5.3 are now $\mathcal{Z}_n = \mathcal{X}_n = \{X_n, \theta_n\}$, $\mathcal{Y}_n = Y_n$ and $\Xi = \emptyset$. Comparing the optimal filter recursion given in (5.12) with the exact optimal filter recursion for the association based NHPP system in (5.11), the densities f and g in (5.12) become

$$\begin{aligned} f(\mathcal{X}_n|\mathcal{Z}_{n-1}) =& p(\theta_n|M_n, \Lambda) p(X_n|X_{n-1}), \\ g(\mathcal{Y}_n|\mathcal{Z}_n) =& p(Y_n|\theta_n, X_n). \end{aligned} \tag{5.40}$$

Since now $\Xi = \emptyset$, we construct our approximate predictive prior $\hat{p}_{n|n-1}(\mathcal{Z}_n)$ according to (5.17) in Section 5.3.3.2, i.e.

$$\hat{p}_{n|n-1}(X_n, \theta_n) = p(\theta_n|M_n, \Lambda) \int p(X_n|X_{n-1}) q_{n-1}^*(X_{n-1}|\Lambda) dX_{n-1}. \tag{5.41}$$

Therefore, according to (5.13) on page 147, the probability law $\hat{p}_n$ for the approximate filtering at the time step n satisfies

$$\hat{p}_n(Y_n, X_n, \theta_n) = p(Y_n|\theta_n, X_n)p(\theta_n|M_n, \Lambda)\hat{p}_{n|n-1}(X_n), \tag{5.42}$$

where the proportional symbol in (5.13) is now replaced by the equality since the normalisation constant can be easily verified to be 1.

5.5.1 Coordinate ascent update

Based on Section 5.3.2, here we assume the factorisation for our mean-field family is $q_n(X_n, \theta_n) = q_n(X_n)q_n(\theta_n)$. The aim of our VB-AbNHPP tracker is to minimise the KL divergence $\mathrm{KL}(q_n(X_n)q_n(\theta_n)||\hat{p}_n(X_n, \theta_n|Y_n))$, or equivalently, to maximise the ELBO in (5.15), which is now

$$\mathcal{F}(q_n) = \mathrm{E}_{q_n(X_n)q_n(\theta_n)} \log \frac{\hat{p}_n(X_n, \theta_n, Y_n)}{q_n(X_n)q_n(\theta_n)}, \tag{5.43}$$

where $\hat{p}_n(X_n, \theta_n, Y_n)$ is given in (5.42). To this end, it iteratively updates $q_n(X_n)$ and $q_n(\theta_n)$ according to (5.16) until convergence. We now derive these updates.

5.5.1.1 Update for $q_n(X_n)$

According to (5.16) on page 148, the update for X_n can be expressed as

$$q_n(X_n) \propto \hat{p}_n(X_n) \exp\left(\mathrm{E}_{q_n(\theta_n)} \log p(Y_n|\theta_n, X_n)\right).$$

By comparing this update with the X_n update in (5.25) from Section 5.4, we can see that these two updates are identical. Moreover, the predictive prior in (5.41) suggests the following predictive prior for X_n:

$$\hat{p}_{n|n-1}(X_n) = \int p(X_n|X_{n-1}) q^*_{n-1}(X_{n-1}|\Lambda) dX_{n-1}, \tag{5.44}$$

which is also identical to the predictive prior in (5.22) from Section 5.4. Consequently, both the prediction and update steps for $q(X_n)$ coincide with (5.28) and (5.29) from Section 5.4.1.1.

5.5.1.2 Update for $q_n(\theta_n)$

The variational distribution $q_n(\theta_n)$ can be updated according to (5.16), (5.5) and (5.6) on page 144, i.e.

$$\begin{aligned} q_n(\theta_n) \propto & p(\theta_n|M_n,\Lambda)\exp\left(\mathrm{E}_{q_n(X_n)}\log p(Y_n|\theta_n,X_n)\right) \\ = & \prod_{j=1}^{M_n} p(\theta_{n,j}|\Lambda)\exp\left(\mathrm{E}_{q_n(X_n)}\log \ell(Y_{n,j}|X_{n,\theta_{n,j}})\right) \\ \propto & \prod_{j=1}^{M_n} q_n(\theta_{n,j}), \end{aligned} \tag{5.45}$$

$$\begin{aligned} q_n(\theta_{n,j}) \propto & p(\theta_{n,j}|\Lambda)\exp\left(\mathrm{E}_{q_n(X_n)}\log \ell(Y_{n,j}|X_{n,\theta_{n,j}})\right) \\ \propto & \frac{\Lambda_0}{V}\delta[\theta_{n,j}=0] + \sum_{k=1}^{K}\Lambda_k l_k \delta[\theta_{n,j}=k], \\ l_k = & \mathcal{N}(Y_{n,j};H\mu_{n|n}^k,R_k)\exp(-0.5\mathrm{Tr}(R_k^{-1}H\Sigma_{n|n}^k H^\top)), \end{aligned} \tag{5.46}$$

where $\mu_{n|n}^k, \Sigma_{n|n}^k$ are given in (5.29), and (5.34) is obtained by substituting (5.7), (5.2) and (5.29).

5.5.2 Initialisation

We suggest an initialisation strategy similar to that in Section 5.4.2: at time step n, the algorithm performs the CAVI by first setting the initial association distribution $q_n^{(0)}(\theta_n)$ as

$$q_n^{(0)}(\theta_n) = \prod_{j=1}^{M_n} q_n^{(0)}(\theta_{n,j}) \tag{5.47}$$

$$\begin{aligned} q_n^{(0)}(\theta_{n,j}) = & \hat{p}_n(\theta_{n,j}|Y_{n,j}) \propto \hat{p}_n(\theta_{n,j},Y_{n,j}) \\ = & \int \hat{p}_n(X_n,\theta_n,Y_n)dY_{n,j-}d\theta_{n,j-}dX_n \\ = & p(\theta_{n,j}|\Lambda)\int \ell(Y_{n,j}|X_{n,\theta_{n,j}})\hat{p}_{n|n-1}(X_n)dX_n \\ \propto & \frac{\Lambda_0}{V}\delta[\theta_{n,j}=0] + \sum_{k=1}^{K}\Lambda_k l_k^0 \delta[\theta_{n,j}=k], \\ l_k^0 = & \mathcal{N}(Y_{n,j};H\mu_{n|n-1}^{k*},H\Sigma_{n|n-1}^{k*}H^\top + R_k). \end{aligned} \tag{5.48}$$

Similar to Section 5.4.2, (5.47) assumes independent initial variational distributions for each $\theta_{n,j}$ to be consistent with the updated form in (5.45). Each $q_n^{(0)}(\theta_{n,j})$ is set to be $\hat{p}_n(\theta_{n,j}|Y_{n,j})$ where $\hat{p}_n$ is the approximate filtering probability law defined in Section 5.3.1 and (5.42). $\hat{p}_{n|n-1}(X_n), \mu_{n|n-1}^{k*}, \Sigma_{n|n-1}^{k*}$ are given in (5.28). Recall that the objective of the CAVI in our setup is to minimise the KL divergence between $q_n(X_n)q(\theta_n)$ and the target distribution $\hat{p}_n(X_n, \theta_n|Y_n)$. The $\hat{p}_n(\theta_{n,j}|Y_{n,j})$ is expected to be a good initialisation for $q_n^{(0)}(\theta_{n,j})$ since it incorporates the information of the corresponding measurement $Y_{n,j}$ under the same probability law (i.e. $\hat{p}_n$) of the target distribution $\hat{p}_n(X_n, \theta_n|Y_n)$.

5.5.3 ELBO computation

With known Poisson rates Λ, the ELBO in (5.43) is now simpler than that in Section 5.4. Similar to Section 5.4.3, here we present the efficient ELBO evaluation that does not require inversions of non-diagonal matrices $\Sigma_{n|n}^k$ and $\Sigma_{n|n-1}^{k*}$. Specifically, if the $q(X_n)$ is the latest updated variational distribution (i.e. the $q(\theta_n)$ has not been updated with the current $q(X_n)$), the ELBO in (5.43) can be exactly expressed as follows,

$$\begin{aligned}
\mathcal{F}(q_n) = &-\frac{1}{2}\sum_{j=1}^{M_n}\sum_{k=1}^{K} q_n(\theta_{n,j}=k)\left(Y_{n,j}^\top R_k^{-1} Y_{n,j} + \log\det R_k\right) \\
&+ \sum_{j=1}^{M_n}\sum_{k=0}^{K} q_n(\theta_{n,j}=k)\log\frac{\Lambda_k}{\Lambda_s q_n(\theta_{n,j}=k)} \\
&+ \frac{1}{2}\sum_{k=1}^{K}\left(\overline{Y}_n^{k\top}\overline{R}_n^{k-1}\overline{Y}_n^k - T_n^{k\top}S_n^{k-1}T_n^k + \log\frac{\det\overline{R}_n^k}{\det S_n^k}\right) \\
&+ \left[\frac{D}{2}\log 2\pi + \log\frac{1}{V}\right]\sum_{j=1}^{M_n} q_n(\theta_{n,j}=0) \\
&- \frac{1}{2}DM_n\log 2\pi,
\end{aligned} \tag{5.49}$$

where, as in Section 5.4.3, $\overline{Y}_n^k, \overline{R}_n^k$ are provided in (5.26) and (5.27), while T_n^k, S_n^k are given in the Kalman filter update (5.29). Note that all these terms have already been calculated for the latest $q(X_n)$ update. D represents the dimension of a single measurement vector $Y_{n,j}$, and V, R_k are defined in (5.2) on page 143. Since the derivation of (5.49) is a simplified version of the process used for (5.37), which can be found in Appendix 5.B, we will not present the derivation here.

In (5.49), the last term is a constant during the current approximate filtering step, so it can be omitted when monitoring the increments of $\mathcal{F}(q_n)$. In typical tracking

Algorithm 10: VB-AbNHPP tracker (with known rate) at time step n

Require: $q^*_{n-1}(X_{n-1}), Y_n, M_n$, maximum iteration limit I, tolerance threshold $\epsilon > 0$.
Output: $q^*_n(X_n), q^*_n(\theta_n)$.
Evaluate $\mu^{k*}_{n|n-1}, \Sigma^{k*}_{n|n-1}$ for $\hat{p}_{n|n-1}(X_n)$ according to (5.28).
for $j = 1, 2, ..., M_n$ **do**
    Evaluate $q^{(0)}_n(\theta_{n,j})$ according to (5.48), and set $q_n(\theta_{n,j}) \leftarrow q^{(0)}_n(\theta_{n,j})$.
end
for $i = 1, 2, ..., I$ **do**
    for $k = 1, 2, ..., K$ **do**
        Evaluate $\overline{R}^k_n, \overline{Y}^k_n$ according to (5.26) and (5.27).
        Compute $\mu^k_{n|n}, \Sigma^k_{n|n}, T^k_n, S^k_n$ according to (5.29) for updating $q_n(X_{n,k})$.
    end
    Evaluate the ELBO $\mathcal{F}^{(i)}_n$ according to (5.49).
    if $\mathcal{F}^{(i)}_n - \mathcal{F}^{(i-1)}_n < \epsilon \wedge i \geq 2$ **then**
        break
    end
    for $j = 1, 2, ..., M_n$ **do**
        Update $q_n(\theta_{n,j})$ according to (5.46).
    end
end
Set $q^*_n(X_n) = \prod_{k=1}^K q_n(X_{n,k})$, and $q^*_n(\theta_n) = \prod_{j=1}^{M_n} q_n(\theta_{n,j})$.

scenarios, both the dynamic transition in (5.9) and the positional measurement function are independent across different coordinates. Consequently, $R_k, \overline{R}^k_n, S^k_n$ are all diagonal matrices, allowing for straightforward computation of all matrix inversions in (5.49).

5.5.4 Algorithm

The approximate filtering algorithm at time step n for our VB-AbNHPP tracker is summarised in Algorithm 10. Many steps in this algorithm can be parallelised, similar to Algorithm 9. Specifically, the evaluation of $q^{(0)}_n(\theta_n)$ and the update of $q_n(\theta_n)$ can be performed in parallel for each association, while the update for $q_n(X_n)$ can be parallelised for each target. Additionally, the computation of the ELBO can be accelerated by utilising parallel computing for the summands in (5.49).

It is worth noting that the pseudo-measurement defined in (5.26) and (5.27) also appears in the well-known PMHT algorithm. The association update in the PMHT algorithm has a similar (but not identical) structure to the variational update derived in (5.46). The primary distinction between the PMHT and our tracker is that PMHT

is developed as a batch algorithm and relies on the EM algorithm, which specifically provides a point estimate for the target state. Although it is theoretically possible to extend the PMHT to an online tracker within a similar approximate filtering framework introduced in Section 5.3.1, its point estimate on the target state may struggle to provide a reliable filtering prior approximation compared to the Gaussian approximation obtained here within the variational Bayes framework.

5.6 Simulation

In this section, we first evaluate the VB-AbNHPP tracker with known rate from Algorithm 10 in simulated scenarios and compare it to other trackers in subsection 5.6.1. Then, in subsection 5.6.2, we briefly demonstrate that the target and clutter's Poisson rates can be accurately estimated using the proposed VB-AbNHPP tracker with unknown rates, as presented in Algorithm 9.

5.6.1 Multi-object tracking with known rate

Here, we compare our VB-AbNHPP tracker in Algorithm 10 with the PF-NHPP tracker [144], the Rao-Blackwellised Gibbs AbNHPP (G-AbNHPP) tracker (Algorithm 2, [50]), and the ET-JPDA filter [147] to demonstrate its efficacy.

Table 5.1 Tracking performance comparisons

		RMSE (mean $\pm 1\sigma$) | track loss percentage (%) | CPU time (s)		
dataset	K	PF-NHPP	G-AbNHPP	ET-JPDA
1	2	7.69±0.64 | 0.00 | 4.05	5.50±0.34 | 0.00 | 0.22	7.49±1.06 | 4.50 | 5e-4
2	4	N/A | 51.8 | 15.9	5.63±0.13 | 0.00 | 0.57	8.40±2.97 | 8.75 | 2e-3
3	10	–	6.06±0.25 | 0.10 | 0.67	9.41±1.99 | 16.3 | 0.01
4	20	–	6.25±0.26 | 0.65 | 2.31	N/A | 22.6 | 0.03

		RMSE (mean $\pm 1\sigma$) | track loss percentage (%) | CPU time (s)	
dataset	K	VB-AbNHPP$^{(1)}$	VB-AbNHPP
1	2	5.64±0.58 | 0.00 | 3e-6	5.51±0.43 | 0.00 | 6e-6
2	4	5.91±0.37 | 0.25 | 6e-6	5.75±0.29 | 0.00 | 1e-5
3	10	6.50±0.50 | 2.20 | 8e-6	6.03±0.32 | 0.70 | 2e-5
4	20	7.31±0.70 | 4.05 | 1e-5	6.30±0.40 | 1.90 | 3e-5

We assume that targets move in a 2D surveillance area with the state in the d-th dimension being $X_{n,k}^d = [x_{n,k}^d, \dot{x}_{n,k}^d]^T$ which contains the target's position and velocity.

Four datasets with the target number increasing from 2, 4, 10, to 20 are generated with each dataset containing 100 Monte Carlo runs with different synthetic trajectories and measurements. For all datasets, we utilise the constant velocity model with a transition density in (5.9), where the transitions in two dimensions are independent and for each d-th dimension, we have

$$F_{n,k}^{d} = \begin{bmatrix} 1 & \tau \\ 0 & 1 \end{bmatrix}, \quad Q_{n,k}^{d} = 25 \begin{bmatrix} \tau^3/3 & \tau^2/2 \\ \tau^2/2 & \tau \end{bmatrix}, \quad B_{n,k}^{d} = 0. \tag{5.50}$$

We set the total time steps to 50, and the time interval between observations is $\tau = 1$s. The target is simulated as an extended target with an elliptical extent which are fixed over time, and its covariance in (5.2) is set to $R_k = 100\mathrm{I}_2, (k = 1, ..., K)$ where I_2 is a 2-D identity matrix. The target rates are set to 5 for all targets in the surveillance area. The clutter density (i.e. Λ_0/V) is set as 10^{-5} per unit area, and correspondingly, the average Λ_0 for datasets 1 to 4 is 24, 50, 93, and 150, respectively. To visualise the tracking scenario of the datasets, in Fig. 5.2, we present one example of measurement data from each dataset. We can see that the clutter is dense and targets frequently come close to each other, making it a challenging scenario for multi-target tracking.

The metrics utilised to measure the tracking performance are the RMSE and the percentage of track loss. First, we define a track as successful if the corresponding target is tracked for at least 85% of its lifespan. A target is considered tracked if the estimated location is within a cut-off distance (50 in this experiment) from the ground truth. The percentage of track loss for each run is computed as the ratio of the number of unsuccessful track(s) and the total number of tracks. Then, the overall track loss performance is evaluated by averaging the track loss percentage over all 100 runs. For the fully tracked runs that have no track loss, we calculate the RMSE and its mean and standard deviation over 100 Monte Carlo runs. Meanwhile, to evaluate the computational complexity, we monitor the CPU time required at a single time step and average it over all time steps (System: Intel(R) Core(TM) i7-8550 CPU@1.80 GHz, 8 GB RAM). The overall CPU time presented in the table is the averaged value across all runs.

The tracking results of all methods are presented in Table 5.1. For the PF-NHPP tracker, we employ 3000 particles in the particle filtering implementation. For the G-AbNHPP tracker, we employ a Rao-Blackwellisation scheme (details can be found in [146, 50]) and implement it using 100 particles with a 40-iteration burn-in time for datasets 1 and 2, and 50 particles with a 20-iteration burn-in time for datasets 3 and 4 to reduce the computational time. To visualise the efficacy of our iterative VB method,

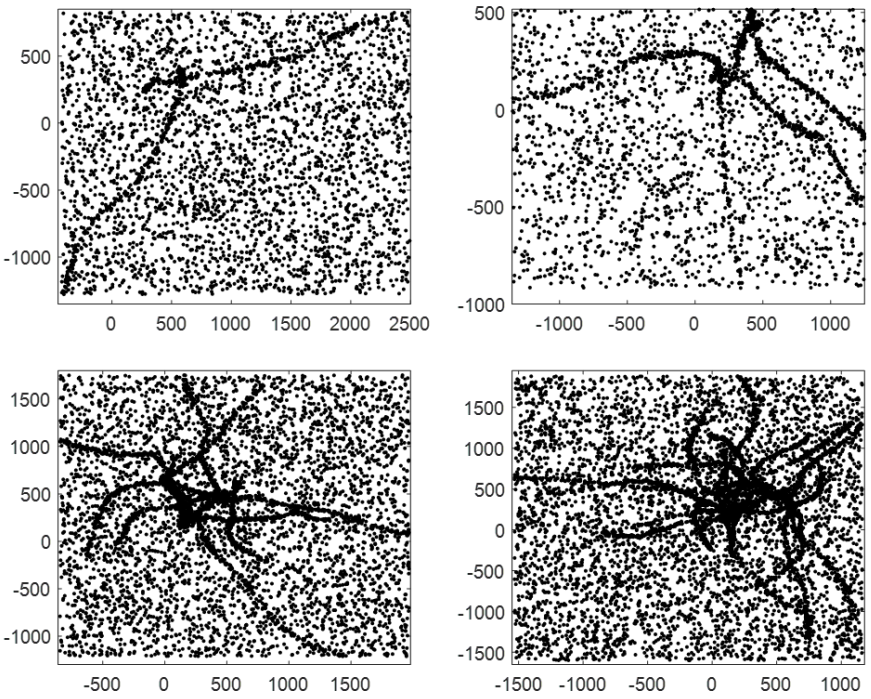


Fig. 5.2 Four examples of measurement data with increasing target and clutter density: from left to right, top to bottom, the number of targets is 2, 4, 10, and 20, and the average Poisson rate is 24, 50, 93, and 150.

we present results for both the VB-AbNHPP$^{(1)}$ tracker that uses only one iteration ($I = 1$ in Algorithm 9), and the standard VB-AbNHPP tracker that performs multiple iterations until convergence. For the standard VB-AbNHPP tracker, the maximum iteration number I and threshold ϵ for ELBO increment in Algorithm 9 are set as 100 and 1e-2. In our experiments, such a setup usually leads to convergence in less than 10 iterations.

From the Table 5.1, we can observe that our method has a promising performance in both time efficiency and tracking accuracy. In terms of efficiency, our VB-AbNHPP tracker has a huge advantage over two sampling-based methods, i.e. the PF-NHPP tracker and the G-AbNHPP tracker; it is also faster than the ET-JPDA tracker, and the difference becomes more evident when the target number increases. As for the tracking performance, the ET-JPDA tracker is much more prone to losing tracks, and in dataset 4 where the target number reaches 20, it has no fully tracked run in all 100 runs. Comparatively, our method significantly outperforms the ET-JPDA with a much lower RMSE and track loss percentage across all datasets. The PF-NHPP tracker, while requiring the longest time of the four methods, has the largest RMSE value in dataset

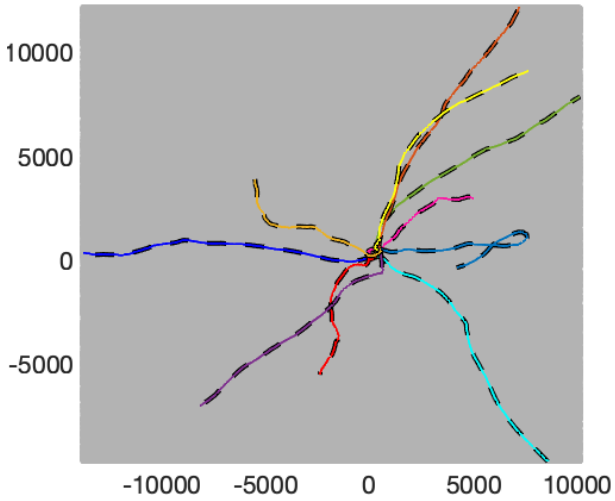


Fig. 5.3 Estimated trajectories (colored lines) of VB-AbNHPP (Algorithm 9) for the tracking and rate estimation task in Section 5.6.2. The black dashed lines are the ground truth.

1, and it fails to track all targets when the target number increases to 4 or larger. Meanwhile, it can be seen that, given an adequate sample size, the G-AbNHPP tracker has the smallest RMSE; however, when the target number is larger such as in datasets 3 and 4, the tracking performance has to be compromised by minimising the sample size so as to track targets in a timely manner. Therefore, compared with the G-AbNHPP tracker, our method can obtain a comparable tracking performance with an advantage in implementation efficiency. Finally, it can be seen that the tracking performance of our VB-AbNHPP tracker is greatly enhanced compared with the VB-AbNHPP$^{(1)}$ that uses only one iteration. It demonstrates that the iterative coordinate ascent updates indeed improve the tracking performance, and the minor extra computational time is worthwhile.

5.6.2 Tracking with rate estimation

In this subsection, we give a simple demonstration of rate estimation with the VB-AbNHPP tracker in Algorithm 9; the target extent can be estimated in a similar fashion if needed. We consider a tracking and rate estimation task where 10 targets move in different directions from initial positions around the origin point. A synthetic dataset is generated for 200 time steps with the system models in Section 5.6.1, and 10 targets' Poisson rates are randomly generated from 1.5 to 10. The ground truth of 10 targets' trajectories over 200 time steps is shown in Fig. 5.3, and the ground truth of each target's Poisson rate is shown in Fig. 5.4. The clutter density is set to 10^{-5} per unit

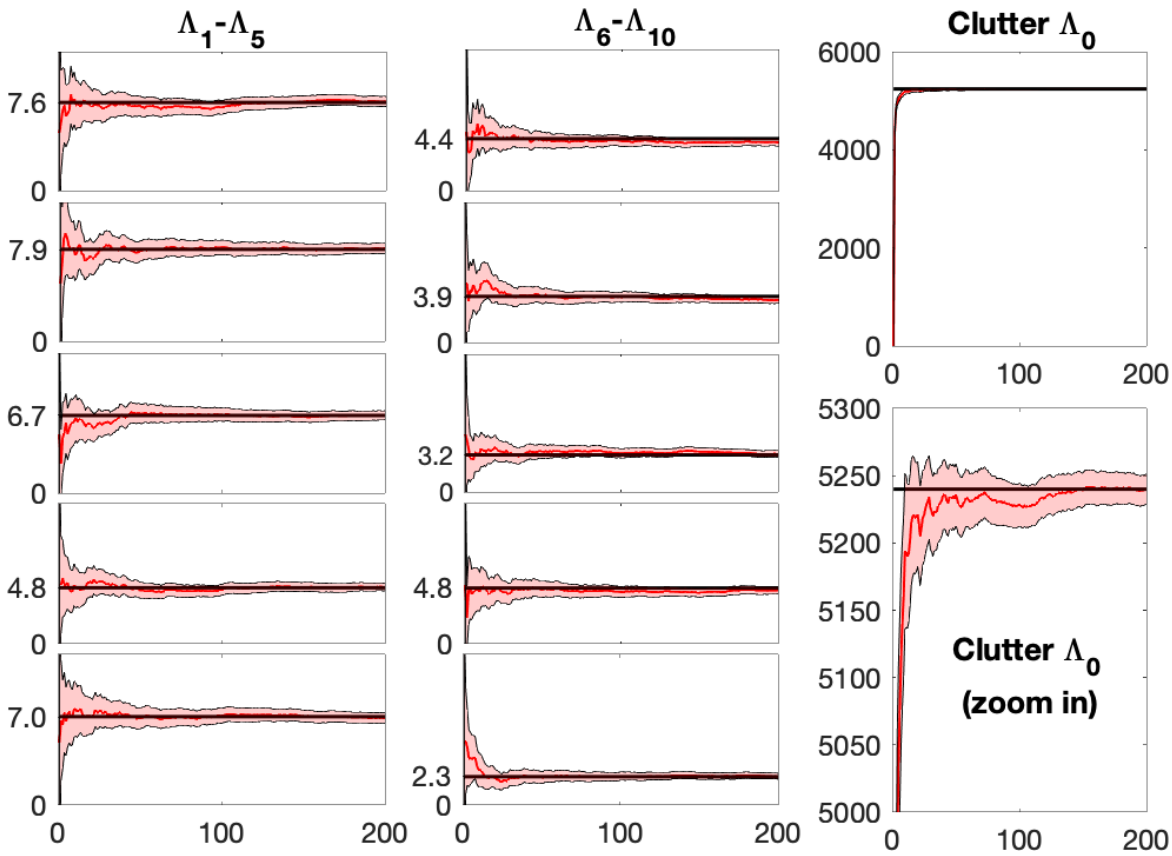


Fig. 5.4 VB-AbNHPP (Algorithm 9) rate estimation over 200 time steps for the tracking and rate estimation task in Section 5.6.2. The ground truth of the Poisson rates of targets and clutter is shown by the black horizontal line in each subfigure. The red curve is the mean of $q_n^*(\Lambda)$, and the light red shade is the confidence region of ± 2 standard deviation of $q_n^*(\Lambda)$.

area, and the clutter rate is 5240 for the considered tasks. The measurements from all time steps are shown as gray dots in Fig. 5.3, appearing as a grey background due to their excessively large number.

We use the VB-AbNHPP in Algorithm 9 to simultaneously track targets and estimate target and clutter rates. In particular, a Gamma distribution with shape parameter 1 and scale parameter 5 is set as the initial Poisson rate prior for all 10 targets and clutter. The γ_n in (5.31) is chosen as $\gamma_n = 1 - 0.1 \times (\max\{1, n-10\})^{-0.9}$, i.e. γ_n is fixed as 0.9 for the first 11 time steps, then strictly increases and approaches 1 when n is large. The choice of $\{\gamma\}_n$ is selected corresponding to Section 5.3.3.2, where $\gamma_{n-1} \in (0, 1]$, and the sequence $\{\gamma\}_n$ should be monotonically increasing. As with our previous experiments, the prior of initial target state is set as the ground truth, and $I = 100, \epsilon = 0.01$ are used to end the iterative updates.

A successful tracking result of VB-AbNHPP is shown in Fig. 5.3, where the estimated trajectories are overlapped with the ground truth. The rate estimation results are shown in Fig. 5.4. As time goes on, the converged variational distribution $q_n^*(\Lambda)$ has a smaller variance and its mean successfully converges to the ground truth rate for all targets and clutter. This agrees with our anticipation in Section 5.3.3.2 that $q_n^*(\Lambda)$ would eventually converge to ground truth Λ as a point estimator. Note that these accurate estimation results for different ground truths of rates are obtained with the identical initial prior and the decreasing sequence $\{\gamma\}_n$. In particular, the ground truth of clutter rate is far from the specified initial prior. This demonstrates the robustness of the rate estimation of VB-AbNHPP under an inaccurate initial prior.

5.7 Conclusion

In this chapter, we propose an online VB-AbNHPP tracker and an adaptive tracker that can simultaneously perform the tracking and parameter learning of the target and clutter Poisson rates, based on a general coordinate ascent variational filtering framework developed here. The developed tracker offers a fast and parallelisable implementation with competitive tracking performance, which makes it promising in large scale target tracking scenarios. While we only derive the VB-AbNHPP tracker with rate estimation, other parameters such as measurement covariances/taregt extents can be learnt in a similar manner. The current dynamic model in the DBN in Section 5.2 can also be replaced by a more complicated intent-driven model (e.g. in Chapter 2 and Chapter 3, [61, 94]) or leader-follower model [142, 152], such that the same framework can still be applied to infer the targets' destination/leader in a clutter scenario. Moreover, a more accurate conditionally factorised variational Bayes described in Chapter 4 and [127] will be considered for more challenging tracking scenarios that involve frequent track coalescence and/or in which target rates are low and in heavy clutter. These extensions will be presented in future work.

Appendix 5.A Evaluation of the initial variational distribution

Here we provide a detailed derivation of the initial variational distribution $\hat{p}_n(\theta_{n,j}|Y_{n,j}, \Lambda = \hat{\Lambda})$ in (5.36). According to the definition of $\hat{p}_n$ in (5.23), we have

$$\hat{p}_n(X_n, \theta_n, \Lambda, Y_n) = p(Y_n|\theta_n, X_n)p(\theta_n|M_n, \Lambda)\hat{p}_{n|n-1}(X_n)p(M_n|\Lambda)\hat{p}_{n|n-1}(\Lambda)/c_M, \quad (5.51)$$

where c_M is a normalisation constant. Then by marginalising Y_n, X_n, θ_n out of (5.51), we have

$$\hat{p}_n(\Lambda) = \sum_{\theta_n} \int \int \hat{p}_n(X_n, \theta_n, \Lambda, Y_n) dY_n dX_n = p(M_n|\Lambda)\hat{p}_{n|n-1}(\Lambda)/c_M, \quad (5.52)$$

where c_M is the same constant in (5.51). Divide (5.51) by (5.52):

$$\begin{aligned} \hat{p}_n(X_n, \theta_n, Y_n|\Lambda) &= p(Y_n|\theta_n, X_n)p(\theta_n|M_n, \Lambda)\hat{p}_{n|n-1}(X_n) \\ &= \hat{p}_{n|n-1}(X_n) \prod_{j=1}^{M_n} \ell(Y_{n,j}|X_{n,\theta_{n,j}})p(\theta_{n,j}|\Lambda), \end{aligned} \quad (5.53)$$

where the last equality follows from (5.5) and (5.6). Denote $Y_{n,j-}$ as all measurements in Y_n except for the $Y_{n,j}$, and $\theta_{n,j-}$ is defined similarly. For any $j = 1, 2, ..., M_n$, by marginalising $Y_{n,j-}, X_n, \theta_{n,j-}$ out of (5.53):

$$\begin{aligned} &\hat{p}_n(\theta_{n,j}, Y_{n,j}|\Lambda) = \sum_{\theta_{n,j-}} \int \int \hat{p}_n(X_n, \theta_n, Y_n|\Lambda) dX_n dY_{n,j-} \\ &= \int \hat{p}_{n|n-1}(X_n)\ell(Y_{n,j}|X_{n,\theta_{n,j}})dX_n \; p(\theta_{n,j}|\Lambda) \prod_{i=1:M_n, i\neq j} \int \ell(Y_{n,i}|X_{n,\theta_{n,i}})dY_{n,i} \sum_{\theta_{n,i}} p(\theta_{n,i}|\Lambda) \\ &= p(\theta_{n,j}|\Lambda) \int \hat{p}_{n|n-1}(X_n)\ell(Y_{n,j}|X_{n,\theta_{n,j}})dX_n \\ &= p(\theta_{n,j}|\Lambda) \left(\frac{1}{V}\delta[\theta_{n,j} = 0] + \sum_{k=1}^{K} l_k^0 \delta[\theta_{n,j} = k] \right) \\ &= \frac{1}{\Lambda_{\text{sum}}} \left(\frac{\Lambda_0}{V}\delta[\theta_{n,j} = 0] + \sum_{k=1}^{K} \Lambda_k l_k^0 \delta[\theta_{n,j} = k] \right). \end{aligned} \quad (5.54)$$

where l_k^0 is defined in (5.36). Note that the integral in the third last line in (5.54) is either a Gaussian marginal likelihood l_k^0 if $\theta_{n,j} \neq 0$, or the $\ell(Y_{n,j}|X_{n,0})$ in (5.2) otherwise (recall that X_n does not include $X_{n,0}$ as assumed Section 5.2). The last line in (5.54) follows from (5.7) and the nature of Kronecker delta function. Finally, (5.36)

is obtained by omitting constant Λ_{sum} in (5.54) and substituting $\hat{\Lambda}_k = \mathrm{E}_{q^*_{n-1}(\Lambda)}\Lambda_k$ to each Λ_k.

Appendix 5.B Derivation of the ELBO for tracking and rate estimation

This appendix derives the ELBO in (5.37) for implementing the VB-AbNHPP tracker with rate estimation. First, we introduce the following Lemma for calculating the summation of quadratic forms:

Lemma 5.B.1. *For symmetric and invertible matrix $C_i \in \mathbb{R}^{D\times D}$, and vectors $x, m_i \in \mathbb{R}^{D\times 1}$ $(i = 1, 2, ..., N)$, we have*

$$\sum_{i=1}^{N} -\frac{1}{2}(x-m_i)^\top C_i^{-1}(x-m_i) = -\frac{1}{2}(x-\mu)^\top \Sigma^{-1}(x-\mu) + \frac{1}{2}\mu^\top\Sigma^{-1}\mu - \frac{1}{2}\sum_{i=1}^{N} m_i^\top C_i^{-1} m_i,$$
$$\Sigma = \left(\sum_{i=1}^{N} C_i^{-1}\right)^{-1}, \qquad \mu = \left(\sum_{i=1}^{N} C_i^{-1}\right)^{-1} \sum_{i=1}^{N} C_i^{-1} m_i. \tag{5.55}$$

A special case often encountered is that the N symmetric matrices C_i differ only in scale, i.e. we have $C_i = C/\omega_i$ for $i = 1, 2, ..., N$. In this case, it is straightforward to derive the following from (5.55)*:*

$$\begin{aligned}
&\sum_{i=1}^{N} -\frac{\omega_i}{2}(x-m_i)^\top C^{-1}(x-m_i) \\
=& -\frac{1}{2}(x-\mu)^\top\Sigma^{-1}(x-\mu) + \frac{1}{2}\mu^\top\Sigma^{-1}\mu - \frac{1}{2}\sum_{i=1}^{N}\omega_i m_i^\top C^{-1} m_i, \qquad (5.56)\\
\Sigma =& \frac{C}{\sum_{i=1}^{N}\omega_i}, \qquad \mu = \frac{\sum_{i=1}^{N}\omega_i m_i}{\sum_{i=1}^{N}\omega_i}.
\end{aligned}$$

Proof. First note that a single quadratic form $(x-\mu)^\top\Sigma^{-1}(x-\mu)$ can always be rewritten as

$$(x-\mu)^\top\Sigma^{-1}(x-\mu) = x^\top\Sigma^{-1}x - x^\top\Sigma^{-1}\mu - (\Sigma^{-1}\mu)^\top x + \mu^\top\Sigma^{-1}\mu. \tag{5.57}$$

Then

$$\begin{aligned}
&\sum_{i=1}^{N} -\frac{1}{2}(x-m_i)^\top C_i^{-1}(x-m_i) \\
=&\sum_{i=1}^{N} -\frac{1}{2}\left[x^\top C_i^{-1}x - x^\top C_i^{-1}m_i - \left(C_i^{-1}m_i\right)^\top x + m_i^\top C_i^{-1}m_i\right] \\
=&-\frac{1}{2}\left[x^\top \sum_{i=1}^{N} C_i^{-1}x - x^\top \sum_{i=1}^{N} C_i^{-1}m_i - \left(\sum_{i=1}^{N} C_i^{-1}m_i\right)^\top x\right] - \frac{1}{2}\sum_{i=1}^{N} m_i^\top C_i^{-1}m_i \qquad (5.58)
\end{aligned}$$

Comparing the terms in the bracket with (5.57), we can find that it matches the quadratic form $-\frac{1}{2}(x-\mu)^\top\Sigma^{-1}(x-\mu)$ in (5.57) if

$$\Sigma^{-1} = \sum_{i=1}^{N} C_i^{-1}, \quad \Sigma^{-1}\mu = \sum_{i=1}^{N} C_i^{-1}m_i, \qquad (5.59)$$

which suggests that Σ, μ being

$$\Sigma = \left(\sum_{i=1}^{N} C_i^{-1}\right)^{-1}, \qquad \mu = \left(\sum_{i=1}^{N} C_i^{-1}\right)^{-1} \sum_{i=1}^{N} C_i^{-1}m_i. \qquad (5.60)$$

Now substitute (5.60), or equivalently (5.59), back to (5.58), we have

$$\begin{aligned}
&\sum_{i=1}^{N} -\frac{1}{2}(x-m_i)^\top C_i^{-1}(x-m_i) \\
=&-\frac{1}{2}\left[x^\top \sum_{i=1}^{N} C_i^{-1}x - x^\top \sum_{i=1}^{N} C_i^{-1}m_i - \left(\sum_{i=1}^{N} C_i^{-1}m_i\right)^\top x\right] - \frac{1}{2}\sum_{i=1}^{N} m_i^\top C_i^{-1}m_i \\
=&-\frac{1}{2}\left[x^\top\Sigma^{-1}x - x^\top\Sigma^{-1}\mu - \left(\Sigma^{-1}\mu\right)^\top x\right] - \frac{1}{2}\sum_{i=1}^{N} m_i^\top C_i^{-1}m_i \\
=&-\frac{1}{2}(x-\mu)^\top\Sigma^{-1}(x-\mu) + \frac{1}{2}\mu^\top\Sigma^{-1}\mu - \frac{1}{2}\sum_{i=1}^{N} m_i^\top C_i^{-1}m_i, \qquad (5.61)
\end{aligned}$$

□

We now begin deriving the ELBO in (5.37). Using (5.22) on page 152, we first rewrite the ELBO in (5.24) as follows

$$\begin{aligned}\mathcal{F}(q_n) =& \mathrm{E}_{q_n(\theta_n)q_n(X_n)} \log \frac{\hat{p}_{n|n-1}(X_n)p(Y_n|\theta_n, X_n)}{q_n(X_n)} \\ &+ \mathrm{E}_{q_n(\Lambda)}\mathrm{E}_{q_n(\theta_n)} \left[\log p(\theta_n|M_n, \Lambda)p(M_n|\Lambda) - \log q_n(\theta_n)\right] \\ &+ \mathrm{E}_{q_n(\Lambda)} \Big[\log \hat{p}_{n|n-1}(\Lambda) - \log q_n(\Lambda)\Big]. \end{aligned} \tag{5.62}$$

Recognise that the last line in (5.62) is $-\mathrm{KL}(q_n(\Lambda)||\hat{p}_{n|n-1}(\Lambda))$ which can be computed as in (5.38). We now compute the second line in (5.62). First using (5.6),(5.7), (5.4), and (5.33), we have

$$\begin{aligned}&\mathrm{E}_{q_n(\theta_n)} \left[\log p(\theta_n|M_n, \Lambda)p(M_n|\Lambda) - \log q_n(\theta_n)\right] \\ =& \log p(M_n|\Lambda) + \mathrm{E}_{q_n(\theta_n)} \sum_{j=1}^{M_n} \Big[\log p(\theta_{n,j}|\Lambda) - \log q_n(\theta_{n,j})\Big] \\ =& \log p(M_n|\Lambda) + \sum_{j=1}^{M_n}\sum_{k=0}^{K} q_n(\theta_{n,j} = k) \log \frac{p(\theta_{n,j} = k|\Lambda)}{q_n(\theta_{n,j} = k)} \\ =& \sum_{j=1}^{M_n}\sum_{k=0}^{K} q_n(\theta_{n,j} = k) \log \frac{\Lambda_k}{q_n(\theta_{n,j} = k)} - \Lambda_{\text{sum}} - \log(M_n!).\end{aligned}$$

Proceeding to compute its expectation with respect to $q_n(\Lambda)$ in (5.32), we obtain the second line in (5.62) as follows

$$\begin{aligned}&\mathrm{E}_{q_n(\Lambda)}\mathrm{E}_{q_n(\theta_n)} \left[\log p(\theta_n|M_n, \Lambda)p(M_n|\Lambda) - \log q_n(\theta_n)\right] \\ =& \sum_{j=1}^{M_n}\sum_{k=0}^{K} q_n(\theta_{n,j} = k)\left(\log \frac{\rho^k_{n|n}}{q_n(\theta_{n,j} = k)} + \psi(\eta^k_{n|n})\right) - \sum_{k=0}^{K} \eta^k_{n|n}\rho^k_{n|n} - \log(M_n!).\end{aligned} \tag{5.63}$$

Finally, we move on to the first line in (5.62). We rewrite it as

$$\begin{aligned}&\mathrm{E}_{q_n(\theta_n)q_n(X_n)} \log \frac{\hat{p}_{n|n-1}(X_n)p(Y_n|\theta_n, X_n)}{q_n(X_n)} \\ =& \mathrm{E}_{q_n(X_n)}\Big[-\log q_n(X_n) + \log \hat{p}_{n|n-1}(X_n) + \mathrm{E}_{q_n(\theta_n)} \log p(Y_n|\theta_n, X_n)\Big].\end{aligned} \tag{5.64}$$

Recall that when $q(X_n)$ is just updated according to (5.25), we have

$$q_n(X_n) = \frac{\hat{p}_{n|n-1}(X_n)\exp\left(\mathrm{E}_{q_n(\theta_n)} \log p(Y_n|\theta_n, X_n)\right)}{\int \hat{p}_{n|n-1}(X_n)\exp\left(\mathrm{E}_{q_n(\theta_n)} \log p(Y_n|\theta_n, X_n)\right) dX_n}, \tag{5.65}$$

where the denominator is the normalisation constant that does not depend on X_n. Substituting (5.65) into (5.64) yields

$$\begin{aligned}
&\mathrm{E}_{q_n(\theta_n)q_n(X_n)} \log \frac{\hat{p}_{n|n-1}(X_n)p(Y_n|\theta_n, X_n)}{q_n(X_n)} \\
=&\mathrm{E}_{q_n(X_n)}\Big[\log \int \hat{p}_{n|n-1}(X_n)\exp\Big(\mathrm{E}_{q_n(\theta_n)} \log p(Y_n|\theta_n, X_n)\Big)\, dX_n\Big] \\
=&\log \int \hat{p}_{n|n-1}(X_n)\exp\Big(\mathrm{E}_{q_n(\theta_n)} \log p(Y_n|\theta_n, X_n)\Big)\, dX_n, \qquad (5.66)
\end{aligned}$$

where the last line follows because the term in the bracket is a constant that does not depend on X_n. Now, to continue to evaluate (5.66), we first compute $\mathrm{E}_{q_n(\theta_n)} \log p(Y_n|\theta_n, X_n)$, which can be expressed as follows using (5.5) and (5.2)

$$\begin{aligned}
&\mathrm{E}_{q_n(\theta_n)} \log p(Y_n|\theta_n, X_n) = \mathrm{E}_{q_n(\theta_n)} \sum_{j=1}^{M_n} \log \ell(Y_{n,j}|X_{n,\theta_{n,j}}) \\
=&\sum_{j=1}^{M_n}\sum_{k=0}^{K} q_n(\theta_{n,j}=k) \log \ell(Y_{n,j}|X_{n,k}) \\
=&\sum_{j=1}^{M_n}\sum_{k=1}^{K} q_n(\theta_{n,j}=k)\left[-\frac{1}{2}(Y_{n,j}-HX_{n,k})^\top R_k^{-1}(Y_{n,j}-HX_{n,k}) - \frac{D}{2}\log 2\pi - \frac{1}{2}\log\det R_k\right] \\
&+\sum_{j=1}^{M_n} q_n(\theta_{n,j}=0)\log\frac{1}{V} \\
=&\sum_{k=1}^{K}\sum_{j=1}^{M_n} -\frac{1}{2}(Y_{n,j}-HX_{n,k})^\top \left(\frac{R_k}{q_n(\theta_{n,j}=k)}\right)^{-1}(Y_{n,j}-HX_{n,k}) + \sum_{j=1}^{M_n} q_n(\theta_{n,j}=0)\log\frac{1}{V} \\
&-\frac{1}{2}\sum_{j=1}^{M_n}\sum_{k=1}^{K} q_n(\theta_{n,j}=k)\log\det R_k - \frac{D}{2}\log 2\pi \sum_{j=1}^{M_n}\sum_{k=1}^{K} q_n(\theta_{n,j}=k). \qquad (5.67)
\end{aligned}$$

Note that we can rewrite the last term in (5.67) as

$$\begin{aligned}
-\frac{D}{2}\log 2\pi \sum_{j=1}^{M_n}\sum_{k=1}^{K} q_n(\theta_{n,j}=k) &= -\frac{D}{2}\log 2\pi \sum_{j=1}^{M_n}(1-q_n(\theta_{n,j}=0)) \\
&= \frac{D}{2}\log 2\pi \sum_{j=1}^{M_n} q_n(\theta_{n,j}=0) - \frac{DM_n}{2}\log 2\pi. \qquad (5.68)
\end{aligned}$$

Also, by using the formula for summation of quadratic terms in (5.56) from Lemma 5.B.1, we have

$$\begin{aligned}
&\sum_{j=1}^{M_n} -\frac{1}{2}(Y_{n,j} - HX_{n,k})^\top \left(\frac{R_k}{q_n(\theta_{n,j}=k)}\right)^{-1} (Y_{n,j} - HX_{n,k}) \\
=& -\frac{1}{2}(\overline{Y}_n^k - HX_{n,k})^\top \overline{R}_n^{k\,-1}(\overline{Y}_n^k - HX_{n,k}) + \frac{1}{2}\overline{Y}_n^{k\,\top}\overline{R}_n^{k\,-1}\overline{Y}_n^k \\
&- \frac{1}{2}\sum_{j=1}^{M_n} q_n(\theta_{n,j}=k) Y_{n,j}^\top R_k^{-1} Y_{n,j}
\end{aligned} \tag{5.69}$$

with $\overline{Y}_n^k, \overline{R}_n^k$ defined in (5.27),(5.26). Now, by substituting both (5.69) and (5.68) back to (5.67), we have

$$\begin{aligned}
&\mathrm{E}_{q_n(\theta_n)} \log p(Y_n|\theta_n, X_n) \\
=& \sum_{k=1}^{K}\left[-\frac{1}{2}(\overline{Y}_n^k - HX_{n,k})^\top \overline{R}_n^{k\,-1}(\overline{Y}_n^k - HX_{n,k}) + \frac{1}{2}\overline{Y}_n^{k\,\top}\overline{R}_n^{k\,-1}\overline{Y}_n^k - \frac{1}{2}\sum_{j=1}^{M_n} q_n(\theta_{n,j}=k) Y_{n,j}^\top R_k^{-1} Y_{n,j}\right] \\
&- \frac{1}{2}\sum_{j=1}^{M_n}\sum_{k=1}^{K} q_n(\theta_{n,j}=k)\log\det R_k + \sum_{j=1}^{M_n} q_n(\theta_{n,j}=0)\log\frac{1}{V} \\
&+ \frac{D}{2}\log 2\pi \sum_{j=1}^{M_n} q_n(\theta_{n,j}=0) - \frac{DM_n}{2}\log 2\pi \\
=& \sum_{k=1}^{K}\left[-\frac{1}{2}(\overline{Y}_n^k - HX_{n,k})^\top \overline{R}_n^{k\,-1}(\overline{Y}_n^k - HX_{n,k}) - \frac{D}{2}\log 2\pi - \frac{1}{2}\log\det \overline{R}_n^k\right] \\
&+ \sum_{k=1}^{K}\left[\frac{D}{2}\log 2\pi + \frac{1}{2}\log\det\overline{R}_n^k\right] - \frac{1}{2}\sum_{j=1}^{M_n}\sum_{k=1}^{K} q_n(\theta_{n,j}=k)\left(Y_{n,j}^\top R_k^{-1} Y_{n,j} + \log\det R_k\right) \\
&+ \frac{1}{2}\sum_{k=1}^{K}\overline{Y}_n^{k\,\top}\overline{R}_n^{k\,-1}\overline{Y}_n^k + \left(\frac{D}{2}\log 2\pi + \log\frac{1}{V}\right)\sum_{j=1}^{M_n} q_n(\theta_{n,j}=0) - \frac{DM_n}{2}\log 2\pi \\
=& \sum_{k=1}^{K}\log\mathcal{N}(\overline{Y}_n^k; HX_{n,k}, \overline{R}_n^k) + C_x,
\end{aligned} \tag{5.70}$$

where C_x is a constant that does not depend on $X_{n,k}$, which is defined as

$$\begin{aligned}
C_x =& \frac{1}{2}\sum_{k=1}^{K}\left(\log\det\overline{R}_n^k + \overline{Y}_n^{k\,\top}\overline{R}_n^{k\,-1}\overline{Y}_n^k\right) - \frac{1}{2}\sum_{j=1}^{M_n}\sum_{k=1}^{K} q_n(\theta_{n,j}=k)\left(Y_{n,j}^\top R_k^{-1} Y_{n,j} + \log\det R_k\right) \\
&+ \frac{DK}{2}\log 2\pi + \left(\frac{D}{2}\log 2\pi + \log\frac{1}{V}\right)\sum_{j=1}^{M_n} q_n(\theta_{n,j}=0) - \frac{DM_n}{2}\log 2\pi
\end{aligned} \tag{5.71}$$

Now substitute (5.70) to (5.66), we have

$$\begin{aligned}
&\log \int \hat{p}_{n|n-1}(X_n) \exp \mathrm{E}_{q_n(\theta_n)} \log p(Y_n|\theta_n, X_n) dX_n \\
=&\log \int \hat{p}_{n|n-1}(X_n) \exp \left(\sum_{k=1}^{K} \log \mathcal{N}(\overline{Y}_n^k; HX_{n,k}, \overline{R}_n^k) + C_x \right) dX_n \\
=&\log \left(\exp C_x \int \prod_{i=1}^{K} \hat{p}_{n|n-1}(X_{n,i}) \prod_{k=1}^{K} \mathcal{N}(\overline{Y}_n^k; HX_{n,k}, \overline{R}_n^k) dX_n \right) \\
=&C_x + \sum_{k=1}^{K} \log \int \hat{p}_{n|n-1}(X_{n,k}) \mathcal{N}(\overline{Y}_n^k; HX_{n,k}, \overline{R}_n^k) dX_{n,k}, \qquad (5.72)
\end{aligned}$$

where the second equality in (5.72) is obtained by using the fact that we always have $\hat{p}_{n|n-1}(X_n) = \prod_{k=1}^{K} \hat{p}_{n|n-1}(X_{n,k})$ for our assumed mean-field family. Note that with a Gaussian $\hat{p}_{n|n-1}(X_{n,k})$ in (5.28), the marginal likelihood in (5.72) can be easily computed in a Gaussian form, i.e.

$$\begin{aligned}
\int \hat{p}_{n|n-1}(X_{n,k}) \mathcal{N}\left(\overline{Y}_n^k; HX_{n,k}, \overline{R}_n^k\right) dX_{n,k} &= \mathcal{N}\left(\overline{Y}_n^k; H\mu_{n|n-1}^{k*}, H\Sigma_{n|n-1}^{k*} H^\top + \overline{R}_n^k\right) \\
&= \exp\left[-\frac{1}{2} T_n^{k\top} S_n^{k-1} T_n^k - \frac{D}{2} \log 2\pi - \frac{1}{2} \log\det S_n^k \right], \qquad (5.73)
\end{aligned}$$

where $\mu_{n|n-1}^{k*}, \Sigma_{n|n-1}^{k*}$ are given in (5.28). T_n^k, S_n^k are defined in (5.29) as by-products of Kalman filter updates. Now substitute (5.73), and (5.71) to the (5.72), we have

$$\begin{aligned}
&C_x + \sum_{k=1}^{K} \log \int \hat{p}_{n|n-1}(X_{n,k}) \mathcal{N}(\overline{Y}_n^k; HX_{n,k}, \overline{R}_n^k) dX_{n,k} \\
=&\sum_{k=1}^{K} \left[-\frac{1}{2} T_n^{k\top} S_n^{k-1} T_n^k - \frac{D}{2} \log 2\pi - \frac{1}{2} \log\det S_n^k \right] + \frac{1}{2} \sum_{k=1}^{K} \left(\log\det \overline{R}_n^k + \overline{Y}_n^{k\top} \overline{R}_n^{k-1} \overline{Y}_n^k \right) \\
&+ \frac{DK}{2} \log 2\pi - \frac{1}{2} \sum_{j=1}^{M_n} \sum_{k=1}^{K} q_n(\theta_{n,j} = k) \left(Y_{n,j}^\top R_k^{-1} Y_{n,j} + \log\det R_k \right) \\
&+ \left(\frac{D}{2} \log 2\pi + \log \frac{1}{V} \right) \sum_{j=1}^{M_n} q_n(\theta_{n,j} = 0) - \frac{DM_n}{2} \log 2\pi \\
=&\left(\frac{D}{2} \log 2\pi + \log \frac{1}{V} \right) \sum_{j=1}^{M_n} q_n(\theta_{n,j} = 0) - \frac{1}{2} \sum_{j=1}^{M_n} \sum_{k=1}^{K} q_n(\theta_{n,j} = k) \left(Y_{n,j}^\top R_k^{-1} Y_{n,j} + \log\det R_k \right) \\
&+ \frac{1}{2} \sum_{k=1}^{K} \left(\overline{Y}_n^{k\top} \overline{R}_n^{k-1} \overline{Y}_n^k - T_n^{k\top} S_n^{k-1} T_n^k + \log \frac{\det \overline{R}_n^k}{\det S_n^k} \right) - \frac{DM_n}{2} \log 2\pi, \qquad (5.74)
\end{aligned}$$

where (5.74) gives the final result of the first line of the ELBO in (5.62), i.e. equation (5.66) that computes $\mathrm{E}_{q_n(\theta_n)q_n(X_n)} \log \frac{\hat{p}_{n|n-1}(X_n)p(Y_n|\theta_n,X_n)}{q_n(X_n)}$, in which $q(X_n)$ is the latest updated variational distribution.

In this way, the complete ELBO in (5.62) is derived with final result given in (5.37), where the first line of equation (5.62) is given in (5.74), the second line of equation (5.62) corresponds to the solution in (5.63), and the last line of equation (5.62) is calculated in (5.38).

Chapter 6

Variational Tracking and Redetection for Closely-spaced Objects in Heavy Clutter

The widely-used NHPP measurement model presented in Section 5.2 allows an object to generate multiple measurements over time. However, it can be difficult to efficiently and reliably track multiple objects under this NHPP model in scenarios with a high density of closely-spaced objects and heavy clutter. While the VB-AbNHPP tracker proposed in the previous chapter provides competitive tracking performance with significant efficiency, extremely challenging tracking scenarios may still lead to track loss. Consequently, this chapter proposes a novel variational localisation strategy, which enables quick rediscovery of missed targets from a large surveillance area under extremely heavy clutter. This strategy is integrated into the standard VB-AbNHPP tracker (Algorithm 10 from Section 5.5), resulting in a robust VB-AbNHPP tracker that can automatically detect and recover from track loss. A more comprehensive literature review on existing multi-object trackers and comparisons with our robust VB-AbNHPP tracker will be conducted. The robust VB-AbNHPP tracker demonstrates significantly better tracking performance than existing trackers in challenging simulated tracking scenes, in terms of both accuracy and efficiency.

This chapter serves as an extension of the previous chapter and is closely related to it. We emphasise that the notation and definitions used in this chapter are consistent with those used in the previous chapter. This chapter includes results that are also part of [153], which is currently under review.

6.1 Introduction

In real-world tracking applications, scenarios are typically complicated and diverse, often presenting challenging situations such as high clutter density, unknown measurement rates, a large number of closely-spaced targets, and occlusion. When tracking targets in such adverse conditions, existing trackers may experience difficulties maintaining a high level of accuracy and efficiency simultaneously, ranging from classical methods such as the joint probabilistic data association (JPDA) filter and multiple hypothesis tracker (MHT) [4], to the most recent techniques including random finite set (RFS) trackers [154, 155] and message passing approaches [149, 156–159].

One major bottleneck is the rapidly growing computational complexity of the data association with the number of targets and measurements. The non-homogeneous Poisson process (NHPP) measurement model [144], which provides an exact association-free measurement likelihood, has recently drawn much attention. The original NHPP tracker [144], however, suffers from the 'curse of dimensionality' due to its particle filter implementation. The same NHPP model was later employed in a JPDA framework [147] to simplify marginal association probabilities; however, it used crude approximations that severely impaired the tracking accuracy (see detailed discussions in Section 5.1). Subsequently, an association-based NHPP (AbNHPP) measurement model was presented in [146, 143], which reintroduces associations into the NHPP model to enable an efficient parallel sampling or a tractable structure. A SMCMC implementation was designed in [146], and for linear Gaussian models, a fast Rao-Blackwellised online Gibbs scheme (here referred to as the G-AbNHPP tracker) was developed in [50] with an enhanced efficiency compared to [146]. Theoretically, these SMCMC methods can converge to optimal Bayesian filters with a large enough sample size, while in practice it can be computationally intensive when target and measurement number are larger. Therefore, we proposed a high-performance AbNHPP tracker based on an efficient variational inference implementation, presented in the previous chapter and in [143]. The devised VB-AbNHPP tracker is verified to achieve comparable tracking accuracy with G-AbNHPP tracker [146] while enjoys a much faster speed.

However, in hostile environments where clutter number in a single time step can be hundreds of times greater than the target's measurements number (see e.g. Fig. 6.1 on page 185 and Fig. 6.8(c) on page 208), all the above-mentioned NHPP/AbNHPP trackers are either computationally impractical [146, 50, 144] or struggle to maintain the tracking accuracy [144, 147, 143]. Therefore, this chapter proposes a novel variational localisation strategy that allows a fast redetection of missed targets from large surveillance area. Embedded with this redetection technique, our upgraded

VB-AbNHPP tracker can automatically detect and recover from the track loss thus providing a robust tracking performance even in extreme cases of heavy clutter with parallelisable implementation and high efficiency.

6.1.1 Related work

The RFS-based trackers, including the Probabilistic Hypothesis Density (PHD) [160] and the Poisson Multiple Bernoulli Mixture (PMBM) filter [154], are amongst the most recent trackers, which provide a compact solution for joint detection and tracking. However, these RFS-based methods often require heuristics and approximations to be feasible in real-world tracking tasks. For instance, the PHD filter can avoid the data association update step, yet it has no closed form solutions and relies on an approximated Gaussian mixture implementation for practical use. Another example is the PMBM filters, which have shown superiority over all other RFS-based trackers [154]. Notwithstanding, all the existing PMBM filters inherit the heuristic pruning, gating and hypothesis management of the MHT framework to limit the exponential increase in the global hypotheses number [154, 155]. Additionally, the most popular implementation in [154] involves further approximation errors: first, it uses the k-best Murty's algorithm to truncate the number of global hypotheses; a pre-processing measurement clustering step is employed for cases that targets generate multiple measurements, e.g. under the NHPP measurement model. Although a sampling-based PMBM filter [155] was devised to reduce the measurement clustering error, the designed MCMC methods are not rigorous and can be computationally intensive. On top of that, it only keeps a truncated subset of data associations and may experience a sharp decline in tracking accuracy under the high data association uncertainty with a large number of targets and clutter.

Alternatively, approximate inference methods [12], such as variational inference [161, 162, 143] and (loopy) belief propagation [148, 149, 156] have been actively investigated due to their promising tracking accuracy and computational efficiency. For example, the belief propagation was used to design a fast data association framework that can maintain a closed-form belief update under a point target measurement model [149]. However, it requires a particle filter implementation when dealing with the tracking problem under the NHPP measurement model [156]; consequently, it loses its computational advantage and proves to be extremely slow in challenging scenes such as in [50] with massive measurement data and heavy clutter. Meanwhile, the cycles in the constructed factor graph raise concerns regarding the convergence and the order of message computation. In contrast, coordinate ascent variational

inference (CAVI) [23, 12], also known as mean field variational inference, is another popular approximate inference method whose convergence can be guaranteed and easily monitored. Compared to MCMC sampling methods, CAVI can typically achieve a comparable performance but with a much more efficient implementation. Several trackers have employed CAVI under a point target measurement model, e.g. [161, 162], whereas the convergence is no longer guaranteed since these trackers have to implement loopy belief propagation to approximate a certain step of variational update. In the previous chapter, we have showed that the original CAVI is sufficient to yield the tractable update without introducing any other approximate inference scheme, for tracking a fixed number object under the NHPP measurement model. Hence, our implementation of CAVI is in the most efficient manner where the convergence is guaranteed and can be easily monitored.

Despite track loss happens frequently in existing multi-object trackers, few literature investigated a solution for retrieving the lost targets in fixed number target tracking. Several primitive track management strategies, including the M/N logic-based method and the sequential probability ratio test (SPRT) [4], have been proposed for track initiation and termination, e.g. in JPDA and MHT. These initiation methods may be used to retrieve lost targets if considering relocating lost targets as detecting a new target birth. More recently, birth processes are employed in many trackers [154, 155, 149, 156, 162] to initiate the track and potentially redetect the missed targets. A major issue that limits their ability to retrieve missed targets under heavy clutter is the low efficiency, since a large number of measurements would lead to massive potential targets, which then require considerable computational efforts to evaluate their existence probability. Although the potential target number can be reduced by pre-processing the measurements (e.g. clustering [156, 154]), when handling target birth under heavy clutter such as Fig. 6.1 and Fig. 6.8(c), to our knowledge, the trackers in [156, 154, 155] are still slow with too many potential targets and fail to reflect the true targets' positions.

6.1.2 Contributions

The first contribution of this chapter is a novel variational object localisation strategy that can efficiently find the potential targets' locations in a large survey area under heavy clutter such as in Fig. 6.1 on page 185 and Fig. 6.8(c) on page 208. With a known target rate and measurement covariance, our strategy can localise the target without an informative positional prior, by using merely a single time step measurements. In particular, this variational localisation strategy proceeds by independently running

multiple CAVIs, each featuring a specifically designed initialiasation that can guide the CAVI to find the most probable target location in a selected small region within the surveillance area. The efficiency of this strategy is then supported by our parallelisable CAVIs and/or a series of wisely selected local regions. Most importantly, this chapter utilises the CAVI in an innovative manner in our localisation strategy. To our knowledge, our localisation strategy is the first attempt to 'control' the CAVI to converge to the desired local optimum, and ultimately to employ this concept to find the global optimum of the considered problem. In contrast, most existing applications (e.g. [51–53, 161, 162]) simply perform the CAVI with a fixed number of iterations and directly adopt the obtained local optimum, whereas this regular routine of CAVI cannot tackle challenging localisation problems in this chapter, since the target's posterior exhibits considerably multi-modal behaviour due to the heavy clutter.

Another major contribution is a VB-AbNHPP tracker with relocation strategy (VB-AbNHPP-RELO). Specifically, we propose a novel track loss detection procedure; once the track loss is detected, the proposed variational localisation strategy can be used to relocate the targets. In this way, our VB-AbNHPP-RELO tracker can robustly track a known number of closely-spaced targets under extremely heavy clutter, allowing the missed targets to be detected and relocated in time automatically and efficiently. Moreover, it enjoys superior tracking accuracy owing to our carefully designed approximate inference paradigm. Compared to other trackers that based on the sampling or maintaining multiple hypotheses (e.g. [50, 156, 154]), our tracker is much faster due to its single Gaussian vector representation of the obtained object state posterior and the efficient CAVI inference implementation; it can be further accelerated due to many parallelisable computational features in the algorithm. Results verify that our proposed VB-AbNHPP-RELO has a significantly higher tracking accuracy with a faster speed over other existing trackers in cases of a large number of closely-spaced targets and heavy clutter.

6.1.3 Layout

The remainder of this chapter is organised as follows. In Section 6.2, we propose and demonstrate a novel variational localisation strategy, which can detect a single target detection in a large surveillance area under heavy clutter. Section 6.3 extends this technique to localise multiple missed targets, based on which and a proposed track loss detection strategy, we develop the VB-AbNHPP-RELO that can timely recover from the track loss. Section 6.4 verifies the performance of both VB-AbNHPP-RELO (with

a known rate) and VB-AbNHPP with rate estimation using simulated data. Finally, Section 6.5 concludes the chapter.

6.2 Variational object localisation under heavy clutter with non-informative prior

First and foremost, we emphasise once again that the notation and definitions used in this chapter are the same as those in the previous chapter. Specifically, the target state X_n, association θ_n, and measurements Y_n, as well as their underlying modelling assumptions and parameters, are all defined in Section 5.2.

This section presents a novel technique for efficiently locating a target amidst a high level of clutter using the variational Bayes framework. Different from tracking scenarios where the previous time step's tracking result can provide an informative prior of the target's position, the localisation strategy discussed here does not require such a strong informative prior. As a result, this technique can be useful for relocating targets once they lose track, or for initialising a tracking algorithm where the target positions are hardly known. Moreover, it has the potential to be developed into a strategy for estimating the number of targets. This section places an emphasis on clarifying the technique's rationale and thus only considers the localisation of a single object. We will extend the relocation technique to handle multiple missed objects in Section 6.3, and integrate it into the complete VB-AbNHPP tracking algorithm.

We assume that the target to be localised follows the NHPP model, and that both the Poisson rates Λ and the measurement covariance are known to us (e.g. trackers have been calibrated and these parameter have been estimated in advance using the proposed method in Section 5.4 of Chapter 5). Our strategy is designed for challenging scenarios where the clutter number in the survey area can be hundreds of times greater than the target's measurement number. We aim to efficiently locate the target only using measurements received at a single time step, and the target can be anywhere in the survey area. Currently, our strategy can handle target Poisson rates as low as 3. In more challenging scenarios where the target Poisson rate is lower, the localisation may be achieved by using measurements from multiple time steps, and this case will be discussed in future.

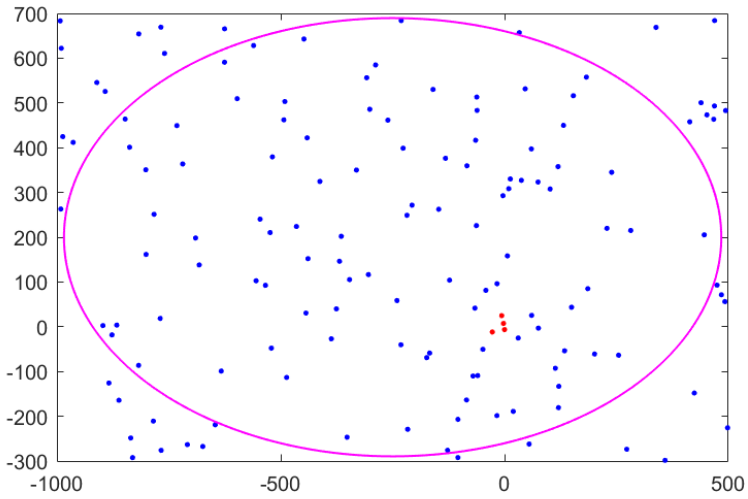


Fig. 6.1 Measurements received at the time step n; ground-truth target position is $(0, 0)$; purple ellipse is the 95% uncertainty ellipse of the target position Gaussian prior; four red dots are target measurements, and the blue dots are clutters.

6.2.1 Problem setting

To clarify the main concept of the proposed procedure, let us consider a typical task of locating a single target under heavy clutter at time step n. This procedure can be performed at the initialisation ($n = 0$) or at any time step ($n \neq 0$) when the track is lost. In this example, the target number $K = 1$, the target state $X_n = X_{n,1}$, and the received measurements Y_n are shown in Fig. 6.1.

We assume that only position measurements of the target state $X_{n,1}$ can be directly observed, with the measurement matrix H and measurement noise covariance matrix $R_{n,1} = 100\mathrm{I}_2$, where I_2 is a $2D$ identity matrix. The target measurement number is assumed to be Poisson distributed with a Poisson rate of $\Lambda_1 = 4$, and these target-oriented measurements are buried in the uniformly distributed clutter with a heavy clutter density of $\Lambda_0/V = 10^{-4}$. The exact target position, i.e. $HX_{n,1}$, is $(0, 0)$. However, we have no other information regarding this position except for a rather flat Gaussian prior $p(X_{n,1})$ and the 95% uncertainty ellipse of position is shown in Fig. 6.1. We denote this ellipse as the survey area, and we will only search for the target within this area. Note that, strictly speaking, $p(X_{n,1})$ should be computed by using the given initial prior $p(X_0)$ and transition $p(X_n|X_{n-1})$. In this section, we directly assign a highly uncertain prior to $p(X_{n,1})$, which is specifically designed for missed targets.

In such scenarios, locating a target can be be difficult due to the dense clutter, which may lead to a multimodal posterior and further confuse the algorithm from finding the true target location. Specifically, when clutter measurements are occasionally densely

displayed in some small regions, these regions become deceptive candidates for the target's true location, leading to several competitive modes in the posterior of the target's position. Taking Fig. 6.1 as an example, it may be difficult to determine whether the target is located at $(0, 0)$ or $(10, 300)$, as both of these two locations have many measurements around them.

6.2.2 Variational localisation strategy

Now we formulate the target localisation task within the variational inference framework, where we aim to approximate the exact posterior $p(X_{n,1}, \theta_n | Y_n, \Lambda)$ with a variational distribution $q(X_{n,1})q(\theta_n)$. Here, the CAVI features similar updates as in (5.29) on page 154 and (5.46) on page 162: the update for $q(X_{n,1})$ is the same as (5.29) except that $\mu^{k*}_{n|n-1}$ and $\Sigma^{k*}_{n|n-1}$ are replaced by the mean and covariance of the prior $p(X_{n,1})$, and the update for $q(\theta_n)$ is the same as (5.46). The standard CAVI with a single initialisation, when applied to the considered task, is prone to getting trapped in local optima and the converged variational distribution only accommodates a single mode of the posterior distribution. To overcome it, the main concept of our localisation technique is to identify multiple competitive modes in the exact posterior $p(X_{n,1}, \theta_n | Y_n, \Lambda)$ by implementing multiple runs of the CAVI in parallel with different initialisations. We then select the most likely mode by evaluating the ELBO calculated for each mode.

Specifically, each run of CAVI starts from initialising the association distribution $q(\theta_n)$ with the $q^{(0)}(\theta_n)$,

$$\begin{aligned} q^{(0)}(\theta_n) =& \prod_{j=1}^{M_n} q^{(0)}(\theta_{n,j}) \\ q^{(0)}(\theta_{n,j}) \propto& \frac{\Lambda_0}{V}\delta[\theta_{n,j} = 0] + \Lambda_1 l_1^0 \delta[\theta_{n,j} = 1], \\ l_1^0 =& \mathcal{N}(Y_{n,j}; m_s, C + R_{n,1}), \end{aligned} \tag{6.1}$$

where N is the total number of initialisations, and m_s $(s = 1, 2, ..., N)$ and C are manually selected for each initialisation and have the same dimensions as $Y_{n,j}$ and $R_{n,1}$, respectively.

Such an initial distribution mimics the form of the initialisation in (5.48), except that $H\mu^{k*}_{n|n-1}$ and $H\Sigma^{k*}_{n|n-1}H^\top$ (i.e. the mean and the covariance of the predictive prior $\hat{p}_{n|n-1}(X_{n,k})$) are replaced by m_s and C, respectively. In a nutshell, the localisation procedure for the considered task is three-step:

Step 1: Choose a C, and assign a series of values for m_s ($s = 1, 2, ..., N$) such that the union of all the 95% error ellipses of $\mathcal{N}(m_s, C)$ can cover the 95% error ellipse of the prior $p(X_{n,1})$. See Fig. 6.2 on page 188 for example.

Step 2: Run CAVI with the initialisation in (6.1) for each pair of m_s and C, then record each $q^*(X_{n,1})$ and its final ELBO. The CAVIs can be run in parallel.

Step 3: Find $q^*(X_{n,1})$ with the highest ELBO, which is the most likely one to capture the true target's position.

In the following, we present two rationales that this localisation method is inherently based upon.

Remark 1. For each s, the CAVI is expected to find the most probable target location (the location with the greatest density of measurements) within the 95% confidence ellipse of $\mathcal{N}(m_s, C)$, with the initialisation in (6.1) and a properly chosen C. See discussions below and in Section 6.2.3.

Remark 2. The ELBO, which equals the negative KL divergence up to an additive constant, reflects the quality of the found variational distribution $q^*(X_{n,1})q^*(\theta_n)$. The higher the ELBO, the better the variational distribution we have found.

Under these two remarks of the proposed localisation strategy, each CAVI for each initialisation will explore a specific local area (e.g. the green circle in Fig. 6.2) to find the 'best' target position in the local area. As the union of these local areas encompasses the survey area, it is anticipated that the target can be successfully localised by a converged variational distribution $q^*(X_{n,1})q^*(\theta_n)$ with the highest ELBO.

Remark 2 is well studied (e.g. in [23, 12]) while Remark 1 is only verified empirically for the considered localisation case in heavy clutter. Particularly, we observe that, typically, Remark 1 holds true only for a small enough C. If C is too large, the CAVI either converges to $q^*(X_{n,1})$ that covers the dense measurements nearest to m_s (i.e. the centre of the local area), or traps in the local optimum with $q^*(X_{n,1}) = p(X_{n,1})$. Note that however, we still would like C to be as large as possible in practical implementation, since this leads to fewer initialisations and hence less computational power required to explore the entire survey area. More details about the empirical properties we observed about the Remark 1, and an informal justification of their rationales is provided in Appendix 6.A.

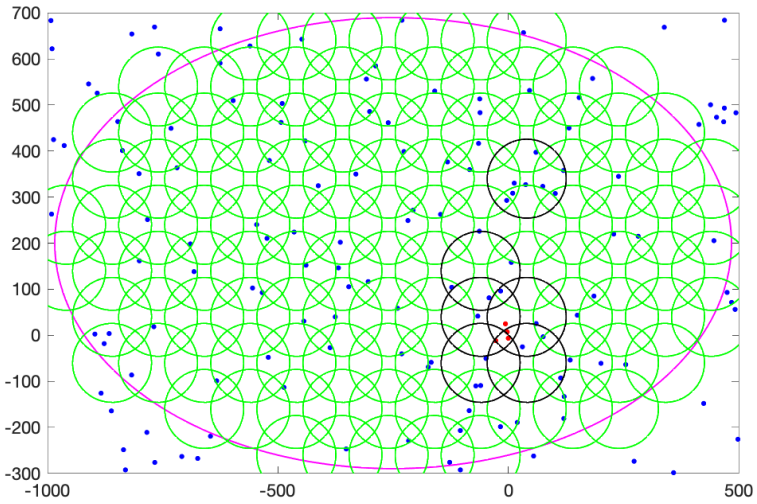


Fig. 6.2 An example of all 95% error ellipses (green/black circles) of $\mathcal{N}(m_s, C)$ used for the initialisation in equation (6.1) (total number $N = 117$). The effects of six of these initialisations (black circles) are further demonstrated in Fig. 6.3.

6.2.3 Demonstration

We now demonstrate the effectiveness of the proposed localisation strategy with an example task. The initialisation setting is shown in Fig. 6.2 where each green circle denotes one initialisation, and all initialisations have the same constant $C = 35^2\mathrm{I}_2$. To show how the algorithm works, we further analyse six initialisations among them, highlighted in black circles in Fig. 6.2, and present the results of the CAVI with these initialisations in Fig. 6.3. In each subfigure in Fig. 6.3, the green circle denotes the initialisation $\mathcal{N}(m_s, C)$; black/grey circles are the target positional variational distribution $q(HX_{n,1})$ evaluated at all iterations, and the color of the circles gradually darkens from grey to black along with the sequence of iterations. Specifically, the lightest grey circle denotes the first iteration's variational distribution and the black circle denotes the converged variational distribution. Note that in Fig. 6.3, we only present the position information $q(HX_{n,1})$ by extracting the mean and covariance of the position from $q(X_{n,1})$. All circles/ellipses in Fig. 6.3 are plotted with a 95% confidence interval. For each subfigure, the ELBO is computed (with an additive constant) by using its converged variational distribution (shown as the black circle). Likewise, we independently carry out the CAVI for all initialisations in Fig. 6.2 until convergence, and compute an ELBO for each initialisation. It turns out that the highest ELBO is 5.671, and the corresponding iterative update results and the initialisation are shown in Fig. 6.3(e). It can be seen from Fig. 6.3(e) that the converged variational

distribution successfully captures the exact target position, which demonstrates that our localisation strategy is effective for this task.

This demonstration example can also facilitate verifying two remarks and the efficacy of the initialisation in (6.1). Let us take a closer look at Fig. 6.3. It can be noticed that, whether or not it converges to the true position, the converged variational distribution $q^*(HX_{n,1})$ is able to capture the most probable location with the greatest density of measurements in the area covered by the green circle, which verifies Remark 1 in Section 6.2.2. In particular, subfigures (a-b) and (e) all locate the target around its exact position $(0, 0)$, and correspondingly achieve the highest three ELBO values, in which case it provides a triple guarantee for the success of our localisation strategy. A noteworthy fact is that the aforementioned deceptive potential target location $(10, 300)$ is found by the initialisation in the subfigure (c) with a relatively high ELBO of 4.31, verifying that the found variational distribution is a competitive mode in the exact posterior. Still, this ELBO is not as high as the ELBO achieved by those variational distributions that capture the exact target location with an ELBO of 5.67. This demonstrates that the ELBO is a reliable metric that can tell the slight difference between multiple competitive modes in the exact posterior, which verifies Remark 2.

Finally, it should be noted that the selection of C in (6.1) is important to the proposed localisation strategy and may need to be tuned for each realisation of the parameter set Λ. Recall that for each $s = 1, 2, ..., N$, if C is too large, the final $q^*(X_{n,1})$ would either concentrate around the measurement nearest to m_s, or be trapped in the local optimum with $q^*(X_{n,1}) = p(X_{n,1})$. For example, the subfigures (b) and (c) in Fig. 6.4 employ the same m_s as in Fig. 6.3(d) and Fig. 6.3(f), respectively, while both having a larger C. It can be seen that both cases in Fig. 6.4 fail to find the most probable target location in the green ellipse. Specifically, even though the green ellipse in Fig. 6.4(b) covers the exact target location, the final $q^*(X_{n,1})$ still finds the local optimum that is closer to m_s. As to the $q^*(X_{n,1})$ in Fig. 6.4(c), it fails to capture the local optimum in Fig. 6.3(f), and instead converges to the prior $p(X_{n,1})$. In contrast, choosing a smaller C leads to a higher computational burden as it requires more initialisations to cover the high confidence region of the prior, and thus more runs of CAVI to explore the entire survey area. Meanwhile, it may be inefficient as each CAVI will only be able to explore a smaller local region. For example, the initialisation in Fig. 6.4(a) has the same m_s as in Fig. 6.3(a) but with a lower C. The final $q^*(X_{n,1})$ captures the most probable target location in the small green ellipse. However, it misses the exact location which would have been covered by a larger C and then captured by the CAVI as in Fig. 6.3(a).

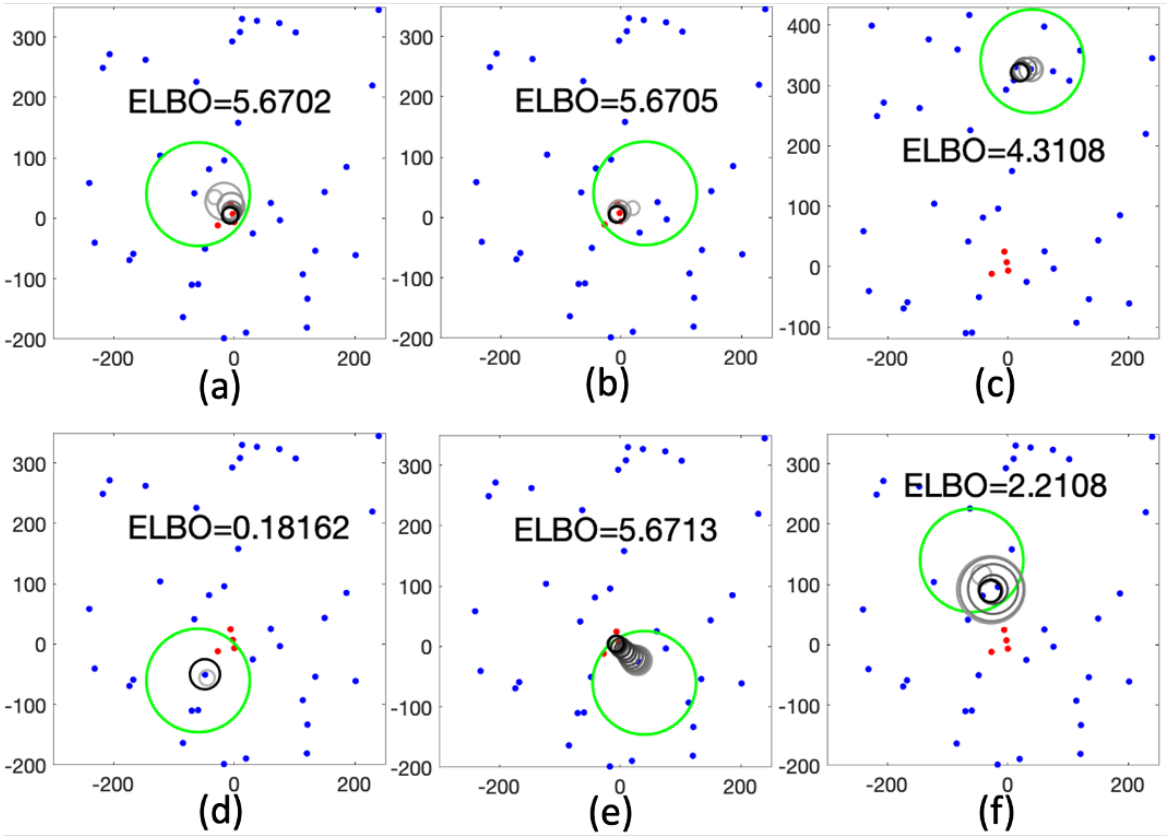


Fig. 6.3 The results of CAVI with six initialisations in Fig. 6.2. The green circle depicts the $\mathcal{N}(m_s, C)$ in (6.1). The grey/black circles represent the iterative updated $q(HX_1)$, whose color gradually darkens with the sequence of iterative updates in CAVI. All circles/ellipses are with 95% confidence interval.

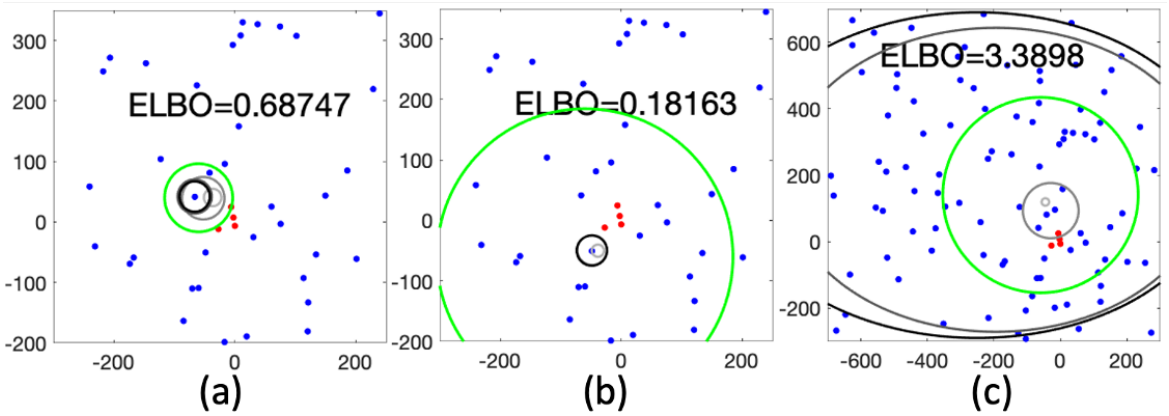


Fig. 6.4 The results of CAVI with three initialisations that employ different m_s and different C.

6.2.4 Acceleration of the localisation strategy

The computational complexity of the presented localisation strategy is primarily determined by the number of runs of CAVI, which is equivalent to the number of initialisations N. In addition to parallelising CAVI runs, enhancing the efficiency of the localisation procedure can be achieved by judiciously reducing N. This reduction can be significant by leveraging our knowledge of the target rate Λ_1 (4 in the current problem setting). Recall that the concept of localisation strategy may be understood by first dividing the entire survey area into N local regions, then employing CAVI to explore each local region. However, some of these local regions are not worth exploring if they include only a few measurements, which result in a total count too low to adhere to our Poisson distribution assumption with rate Λ_1. Hence we can ignore such initialisations to accelerate the algorithm.

For example, the regions that include 0 measurement are obviously not worth exploring; and even if we did explore them, without any measurement, there wouldn't be any reasonable localisation result. In particular, we can perform the CAVI only for the local regions that include at least M^{init} measurements. The choice of M^{init} will be discussed in Appendix 6.B.2 for our full tracking algorithm. Subsequently, the number of eligible initialisations (or green circles) in Fig. 6.2 is $N = 117$ if $M^{init} = 0$. This number reduces to 104, 70 and 36 if M^{init} is set to 1, 2, and 3 respectively. For this specific task, we can still successfully localise the target even if we set $M^{init} = 6$, which only requires 6 initialisations in total.

6.3 VB-AbNHPP tracker with missed objects relocation

We now extend the localisation strategy for a single object in Section 6.2.2 to enable the relocation of multiple lost track objects. This strategy will be employed in the developed VB-AbNHPP tracker, and the procedure of the full VB-AbNHPP tracker with relocation will be given in subsection 6.3.3. Recall that the standard VB-AbNHPP tracker aims to track K targets with labels $1, 2, ..., K$. Now define $\mathcal{L}_n$ as the set of labels for targets whose tracks have been lost after a standard tracking procedure, followed by a track loss detection step in Section 6.3.1 at time step n (detailed steps to obtain $\mathcal{L}_n$ will be described in Algorithm 12). The objective of the proposed relocation strategy at time step n is to relocate these targets before we implement the next tracking procedure at $n + 1$.

Specifically, to relocate multiple objects in $\mathcal{L}_n$, we first assign a non-informative Gaussian prior $\tilde{p}_n(X_{n,h})$, $h \in \mathcal{L}_n$ to each of them. As in Section 6.2, we assume that the Poisson rate Λ is known or has been estimated. Then the target distribution for the relocation task at time step n is

$$\tilde{p}_n(X_n, \theta_n|Y_n) \propto \prod_{h\in\mathcal{L}_n} \tilde{p}_n(X_{n,h})\hat{p}_{n|n-1}(X_{n,k\notin\mathcal{L}_n})p(Y_n|\theta_n, X_n)p(\theta_n|M_n, \Lambda), \tag{6.2}$$

where $X_{n,k\notin\mathcal{L}_n}$ denotes the state of all targets that are not in $\mathcal{L}_n$, and their predictive prior $\hat{p}_{n|n-1}(X_{n,k\notin\mathcal{L}_n})$ has been defined in (5.22) and computed in the standard tracker via (5.28). This target distribution $\tilde{p}_n(X_n, \theta_n|Y_n)$ is defined similarly to $\hat{p}_n(X_n, \theta_n|Y_n)$, where $\hat{p}_n$ is defined in (5.42), except that the predictive priors for missed targets' states are replaced by a non-informative prior $\tilde{p}_n(X_{n,h})$, $h \in \mathcal{L}_n$.

Subsequently, the relocation task can be formulated within the variational Bayes framework as follows: We aim to minimise the KL divergence between the variational distribution $q_n(\theta_n)q_n(X_n)$ and the target distribution $\tilde{p}_n(X_n, \theta_n|Y_n)$ in (6.2). Different from the tracking tasks in Section 5.4 where $q_n(X_n)$ are updated independently for each target, here for the efficiency of the relocation task, we are only interested in updating the $q_n(X_{n,h})$ for missed targets $h \in \mathcal{L}_n$. Moreover, we will localise each target in $\mathcal{L}_n$ one at a time. This is to avoid the exponentially increasing number of (multi-dimensional) local regions that are required to cover the entire multi-dimensional survey area if multiple objects are localised at the same time. For example, if for each single target, 100 local regions are required to overlap the single-dimensional survey area, then it only requires 100×3 initialisations of CAVI in total to locate three targets one at a time; however, it needs 100^3 initialisations (each in a three-dimensional space) to cover the whole survey area in three-dimensional space if three targets are localised together.

It should be noted that when we localise the target h ($h \in \mathcal{L}_n$), other targets (including other targets in $\mathcal{L}_n$ and all targets that are tracked properly) have fixed variational distributions $q_n^*(X_{n,h-})$, where $X_{n,h-}$ denotes all the target states in X_n except $X_{n,h}$. This converged variational distribution $q_n^*(X_n)$ is first obtained from the standard tracking procedure at time step n, and should be updated timely as $q_n^{*new}(X_{n,h})q_n^*(X_{n,h-})$ once a lost track target h in $\mathcal{L}_n$ has been successfully relocated by a converged variational distribution $q_n^{*new}(X_{n,h})$. Subsequently, for each $h \in \mathcal{L}_n$, the CAVI is implemented to maximise the following ELBO by iteratively updating $q_n(\theta_n)$

and $q_n(X_{n,h})$,

$$\begin{aligned}
&\mathcal{F}_{n,h}(q_n(\theta_n), q_n(X_{n,h})) \\
=&\mathrm{E}_{q_n(\theta_n)q_n(X_{n,h})q_n^*(X_{n,h-})} \log \frac{p(Y_n|\theta_n, X_n)p(\theta_n|M_n, \Lambda)\hat{p}_{n|n-1}(X_{n,k\notin\mathcal{L}_n}) \prod_{i\in\mathcal{L}_n} \tilde{p}_n(X_{n,i})}{q_n(\theta_n)q_n(X_{n,h})q_n^*(X_{n,h-})} \\
=&-\mathrm{KL}(q_n(\theta_n)||p(\theta_n|M_n, \Lambda)) - \mathrm{KL}(q_n(X_{n,h})||\tilde{p}_n(X_{n,h})) \\
&+\mathrm{E}_{q_n(\theta_n)q_n(X_{n,h})q_n^*(X_{n,h-})} \log p(Y_n|\theta_n, X_n) + c,
\end{aligned} \tag{6.3}$$

where c is a constant that does not depend on $q_n(\theta_n)$ or $q_n(X_{n,h})$. The specific form of (6.3) is derived in Appendix 6.C. This optimisation is still equivalent to minimising the $\mathrm{KL}(q_n(\theta_n)q_n(X_n)||\tilde{p}_n(X_n, \theta_n|Y_n))$, only now $q_n(X_n)$ is set as $q_n(X_{n,h})q_n^*(X_{n,h-})$ and $q_n^*(X_{n,h-})$ is fixed when localising the target h.

By fixing other targets' variational distributions, the variational localisation of the target h naturally takes into account the impact of other successfully localised targets. Specifically, the measurements that are covered by the fixed variational distributions $q_n^*(X_{n,h-})$ naturally have a considerable probability to associate with the successfully tracked targets in $X_{n,h-}$. In order to achieve a high ELBO, typically $q_n(X_h)$ tends to encompass some dense measurements that are away from other targets, instead of relocating the target h around other properly tracked targets to compete for the association probability of the measurements nearby. Therefore, multiple objects can be relocated to different potential locations, rather than occupying the same place with limited measurements around them.

The optimisation of the ELBO $\mathcal{F}_{n,h}$ in (6.3) requires similar variational updates as in Section 6.2.2. The update for $q_n(X_{n,h})$ is the same as in (5.29) except replacing the $\mu_{n|n-1}^{h*}$ and $\Sigma_{n|n-1}^{h*}$ with the mean and covariance in $\tilde{p}_n(X_{n,h})$. Recall that $q_n^*(X_{n,k}) = \mathcal{N}(\mu_{n|n}^{k*}, \Sigma_{n|n}^{k*})$ and $q_n(X_{n,h}) = \mathcal{N}(\mu_{n|n}^{h}, \Sigma_{n|n}^{h})$, the update of $q_n(\theta_n)$ is also similar to (5.46):

$$\begin{aligned}
q_n(\theta_n) &= \prod_{j=1}^{M_n} q_n(\theta_{n,j}), \\
q_n(\theta_{n,j}) &\propto \frac{\Lambda_0}{V}\delta[\theta_{n,j} = 0] + \sum_{k=1}^{K} \Lambda_k l_k \delta[\theta_{n,j} = k], \\
l_k &= \begin{cases} \dfrac{\mathcal{N}(Y_{n,j}; H\mu_{n|n}^{k*}, R_k)}{\exp(0.5\mathrm{Tr}(R_k^{-1}H\Sigma_{n|n}^{k*}H^\top))}, & k \neq h \\ \dfrac{\mathcal{N}(Y_{n,j}; H\mu_{n|n}^{h}, R_h)}{\exp(0.5\mathrm{Tr}(R_h^{-1}H\Sigma_{n|n}^{h}H^\top))}, & k = h \end{cases}
\end{aligned} \tag{6.4}$$

Moreover, to relocate the lost track target h under heavy clutter, it is essential to implement the same variational localisation strategy with multiple initialisations as presented in Section 6.2.2. That is, we need to first determine an initialisation covariance C, then assign a series values of m_s ($s = 1, 2, ..., N$) such that the union of high (e.g. 95%) confidence ellipses of $\mathcal{N}(m_s, C)$ can cover the high confidence region of our prior $\tilde{p}_n(X_{n,h})$. Then, multiple CAVIs can be carried out in parallel to search for the most likely target position in a local region $\mathcal{N}(m_s, C)$ for each initialisation:

$$\begin{aligned} q_n^{(0)}(\theta_n) &= \prod_{j=1}^{M} q_n^{(0)}(\theta_{n,j}) \\ q_n^{(0)}(\theta_{n,j}) &\propto \frac{\Lambda_0}{V}\delta[\theta_{n,j} = 0] + \sum_{k=1}^{K} \Lambda_k l_k^0 \delta[\theta_{n,j} = k], \\ l_k^0 &= \begin{cases} \mathcal{N}(Y_{n,j}; H\mu_{n|n-1}^{k*}, H\Sigma_{n|n-1}^{k*}H^\top + R_k), & k \neq h \\ \mathcal{N}(Y_{n,j}; m_s, C + R_h), & k = h \end{cases} \end{aligned} \tag{6.5}$$

where $s = 1, 2, ..., N$; $\mu_{n|n-1}^{k*}$ and $\Sigma_{n|n-1}^{k*}$ are the mean and covariance of $\hat{p}_n(X_{n,k})$, which have been computed by (5.28) in the standard tracking procedure. Similar to the initialisation (6.1) in Section 6.2.2, (6.5) mimics the form of (5.48) with a modification to the term l_h^0. As discussed in Section 6.2.4, not all of these initialisations are necessary to implement. We will only consider the initiasations in which the 95% confidence ellipses of $\mathcal{N}(m_s, C)$ include at least M_k^{init} measurements, where M_k^{init} is an eligible initialisation threshold. The choice of M_k^{init} will be discussed in Appendix 6.B.2.

Finally, the relocation strategy of lost track targets for VB-AbNHPP tracker can be summarised in Algorithm 11. In brief, for each missed target, it first 1) determines the uninformative Gaussian prior and settings for N initialisations; then 2) carries out multiple CAVI with all eligible initialisations as discussed above, and finds the converged variational distribution with the highest ELBO; 3) if the found variational distribution is convincing enough, use it as the obtained posterior of this missed target; otherwise give up the relocation for this target.

A final remark on our relocation strategy is that Algorithm 11 does not always relocate all targets in $\mathcal{L}_n$ at every time step n. Rather, it only relocates the lost track target when it is confident about its position. This is because even though the obtained variational distribution that achieves the highest ELBO is expected to find the most likely target's location, it may still not capture the true position. If we relocate a target to the wrong position, the algorithm may take many time steps to realise this relocation is wrong and the target may be lost for too long, which may further

Algorithm 11: Relocation strategy at time step n

Require: $Y_n, M_n, I, \epsilon, \hat{p}_{n|n-1}(X_n), q_n^*(X_n)$ from the standard tracker, the exact or estimated Λ, missed target set $\mathcal{L}_n$ (obtained in Algorithm 12 when this algorithm is invoked), relocation initialisation thresholds $M_{h\in\mathcal{L}_n}^{reloc}$, and eligible initialisation thresholds $M_{h\in\mathcal{L}_n}^{init}$.

Output: Successfully tracked/relocated set $\mathcal{K}_n$, refined $q_n^*(X_n)$ and $q_n^*(\theta_n)$.

Initialise $\mathcal{K}_n = \{k \in \{1, 2, ..., K\} : k \notin \mathcal{L}_n\}$.

foreach $h \in \mathcal{L}_n$ **do**

  Assign an uninformative Gaussian prior $\tilde{p}_n(X_{n,h})$.

  Determine $m_s (s = 1, 2, ..., N)$ and C for initialisations (6.5) such that the union of 95% confidence ellipse of $\mathcal{N}(m_s, C)$ covers $\tilde{p}_n(X_{n,h})$.

  for $s = 1, 2, ..., N$ **do**

    if there are less than M_h^{init} measurements in the 95% confidence ellipse of $\mathcal{N}(m_s, C)$ **then**

      Set $\mathcal{F}_{n,h}^s = -\infty$.

      continue

    end

    Evaluated $q_n^{(0)}(\theta_n)$ via (6.5), and initialise $q_n(\theta_n) = q_n^{(0)}(\theta_n)$.

    for $i = 1, 2, ..., I$ **do**

      Evaluate $\overline{R}_n^h, \overline{Y}_n^h$ according to (5.26)(5.27).

      Update $q_n(X_{n,h})$ via (5.29) where $\mu_{n|n-1}^{k*}, \Sigma_{n|n-1}^{k*}$ are replaced by the mean and covariance of $\tilde{p}_n(X_{n,h})$.

      Evaluate the ELBO $\mathcal{F}_{n,h}^{(i)}$ according to (6.3).

      if $\mathcal{F}_{n,h}^{(i)} - \mathcal{F}_{n,h}^{(i-1)} < \epsilon \wedge i \geq 2$ **then**

        break

      end

      Update $q_n(\theta_n)$ according to (6.4).

    end

    Set $\mathcal{F}_{n,h}^s = \mathcal{F}_{n,h}^{(i)}$, and $q_n^s(X_{n,h}) = q_n(X_{n,h})$.

  end

  Find the best index $w = \arg\max_s \mathcal{F}_{n,h}^s$.

  if $\sum_{j=0}^{M_n} q_n^w(\theta_{n,j} = h) \geq M_h^{reloc}$ **then**

    Update $q_n^*(X_n) \leftarrow q_n^w(X_{n,h}) q_n^*(X_{n,h-})$, i.e. update $\mu_{n|n}^{h*}$, $\Sigma_{n|n}^{h*}$ as the mean and covariance of $q_n^w(X_{n,h})$.

    Update $\mathcal{K}_n \leftarrow \mathcal{K}_n \bigcup \{h\}$.

  else

    Update $q_n^*(X_n) \leftarrow \tilde{p}_n(X_{n,h}) q_n^*(X_{n,h-})$, i.e. update $\mu_{n|n}^{h*}$, $\Sigma_{n|n}^{h*}$ as the mean and covariance of $\tilde{p}_n(X_{n,h})$.

  end

end

Update $q_n^*(\theta_n)$ with the refined $q_n^*(X_n)$, i.e. using (6.4) where $\mu_{n|n}^h, \Sigma_{n|n}^h$ are replaced by $\mu_{n|n}^{h*}, \Sigma_{n|n}^{h*}$.

deteriorate the tracking performance. Therefore, Algorithm 11 will first determine whether the obtained variational distribution is convincing enough by using an effective relocation criterion. If this criterion is satisfied, the algorithm refines $q_n^*(X_n)$ with the obtained variational distribution; if not, the prior $\hat{p}(X_{n,h})$ will be employed to update the $q_n^*(X_n)$. Such an effective relocation criterion, which has already been specified in Algorithm 11, will be explained in Appendix 6.B.2.

In the following subsections, we will first describe the track loss detection strategy, and then explain the rationale of the effective relocation criterion in Algorithm 11. Finally, we will apply all these techniques to develop the full VB-AbNHPP-RELO tracker in Section 6.3.3.

6.3.1 Track loss detection

The proposed relocation strategy (Algorithm 11) described above is only triggered when a track loss is detected. To this end, the (estimated) number of measurements produced by a target is used to monitor whether the target is tracked properly. The exact number of measurements $M_{n,k}$ generated at time step n by the target k can be defined with the association θ_n, i.e. $M_{n,k} = \sum_{j=0}^{M_n} I(\theta_{n,j} = k)$. A convenient point estimate of this target measurement number can be computed as a byproduct of the VB-AbNHPP using the converged variational distribution $q_n^*(\theta_n)$, i.e.

$$\hat{M}_{n,k} = \mathrm{E}_{q_n^*(\theta_n)} M_{n,k} = \sum_{j=0}^{M_n} q_n^*(\theta_{n,j} = k). \tag{6.6}$$

If the algorithm fails to track the target k, the estimated target position may have fallen in an area with no real measurements but only sporadic clutter; in this case, $\hat{M}_{n,k}$ is expected to be very small as there may be few measurements associated with target k. Therefore, a criterion for detecting a track loss event is that $\hat{M}_{n,k}$ is too low.

Specifically, we will record $\hat{M}_{t,k}$ for each time step t and each target k, and if the total (estimated) number of measurements generated during τ_k ($\tau_k \geq 1$) time steps before the current time step n falls into a certain positive threshold M_k^{los}, in other words, if the following event E happens,

$$\text{event } E: \sum_{t=n-\tau_k+1}^{n} \hat{M}_{t,k} \leq M_k^{los}, \tag{6.7}$$

we conclude that the track of the target k has been lost. The details about the choice of the parameters τ_k and $\hat{M}_{n,k}$, including an automatic parameter selection strategy, are presented in Appendix 6.B.1.

6.3.2 Effective relocation criterion

Similar to the track lost detection in Section 6.3.1, the (estimated) number of measurements produced by a relocated target is used to determine whether the obtained variational distribution is convincing enough. Specifically, the estimated number of measurements generated from the target $h \in \mathcal{L}_n$ based on the obtained variational distribution $q_n^w(X_{n,h}, \theta_n)$ is $\sum_{j=0}^{M_n} q_n^w(\theta_{n,j} = h)$. The effective relocation criterion in Algorithm 11 is satisfied when this estimated measurement number is greater than the relocation threshold M_h^{reloc}, i.e.

$$\sum_{j=0}^{M_n} q_n^w(\theta_{n,j} = h) \geq M_h^{reloc} \tag{6.8}$$

The reason is that a high estimated measurement number is more likely to have been generated by the target itself, than by uniformly distributed clutter that happened to cluster in a small area by chance. As a result, a high estimated measurement number makes it more likely that the obtained variational distribution $q_n^w(X_{n,h})$ has successfully relocated the target h. The selection of the relocation threshold M_h^{reloc} will be discussed in Appendix 6.B.2.

6.3.3 Full VB-AbNHPP tracker with relocation Strategy

Finally, we are in a position to present the complete VB-AbNHPP tracker with relocation (VB-AbNHPP-RELO) algorithm. For simplicity, we only present the VB-AbNHPP-RELO tracker for tasks where the Poisson rate Λ and measurement covariance R_k $(k = 1, 2, ...K)$ are known or have been accurately estimated. This allows our algorithm to robustly track closely-spaced targets under extremely heavy clutter. In this setup, the track loss detection parameters τ_k, M_k^{los}, relocation threshold M_k^{reloc}, and eligible initialisation threshold M_k^{init} $(k = 1, 2, ..., K)$ can be easily and automatically selected according to Appendix 6.B. These parameters, along with initialisation covariance C and convergence monitoring parameters I, ϵ, can all be determined before performing tracking tasks and are not required to be updated in subsequent time steps. For convenience, all these parameters will not be listed as inputs in our single time step VB-AbNHPP-RELO procedure summarised in Algorithm

Algorithm 12: Tracking with relocation at time step n

Require: $q^*_{n-1}(X_{n-1}), Y_n, M_n$, successfully tracked/relocated set $\mathcal{K}_{n-1}$. If $\tau_k \geq 2$, also require $\hat{M}_{t,k}$ ($k \in \mathcal{K}_{n-1}$ and t from $n-\tau_k+1$ to $n-1$).
Output: $q^*_n(X_n), \mathcal{K}_n$. If $\tau_k \geq 2$, also output $\hat{M}_{t,k}$ for all $k \in \mathcal{K}_n$ and t from $n-\tau_k+2$ to n.
Run the standard tracker with known Λ, i.e. Algorithm 10 from Section 5.5, to obtain $\hat{p}_{n|n-1}(X_n), q^*_n(X_n)$, and $q^*_n(\theta_n)$.
Initialise $\mathcal{L}_n = \{k \in \{1, 2, ..., K\} : k \notin \mathcal{K}_{n-1}\}$.
foreach $k \in \mathcal{K}_{n-1}$ **do**
    Evaluate $\hat{M}_{n,k}$ according to (6.6).
    if $\sum_{t=n-\tau_k+1}^{n} \hat{M}_{t,k} \leq M_k^{los}$ **then**
        Update $\mathcal{L}_n \leftarrow \mathcal{L}_n \bigcup \{k\}$.
    end
end
if $\mathcal{L}_n \neq \emptyset$ **then**
    Run the relocation procedure (i.e. Algorithm 11) to obtain $\mathcal{K}_n$ and the refined $q^*_n(X_n)$ and $q^*_n(\theta_n)$.
    foreach $k \in \mathcal{K}_n$ **do**
        Update $\hat{M}_{n,k}$ via (6.6) with the refined $q^*_n(\theta_n)$.
        if $k \in \mathcal{L}_n \wedge \tau_k \geq 3$ **then**
            Set $\hat{M}_{t,k} = \Lambda_k$ for t from $n-\tau_k+2$ to $n-1$.
        end
    end
end

12. Note that in Algorithm 12, the standard VB-AbNHPP tracker with known Λ and R_k is incorporated, which has been outlined in Algorithm 10 from Section 5.5. In a nutshell, Algorithm 12 first 1) performs the standard VB-AbNHPP tracker for the current time step, then 2) updates the missed target set $\mathcal{L}_n$ by track loss detection, and finally 3) implements the relocation only for missed targets in $\mathcal{L}_n$.

For targets relocated successfully at the current time step, their estimated measurement numbers at previous time steps are in principle unavailable to us. However, the track loss detection in (6.7) requires the estimated measurement number from previous τ_k time steps. Thus, in order to ensure the track loss detection can be directly carried out in the next time step, Algorithm 12 automatically sets the estimated measurement number at previous time steps as target's Poisson rates. This empirical setup allows for the timely detection of track loss for any properly tracked target at any time step.

It is yet be discussed the choice of the uncertain Gaussian prior $\tilde{p}_n(X_{n,h})$ for the relocation of the missed target $h \in \mathcal{L}_n$ in Algorithm 12 and Algorithm 11. Typically,

the mean of $\tilde{p}_n(X_{n,h})$ should be set to the target h's latest state estimate when it was properly tracked (for at least 1 or 2 time steps). Moreover, it is practically efficient to set a relatively small covariance of $\tilde{p}_n(X_{n,h})$ for the target h whose track has just been lost, e.g. for $h \in \mathcal{L}_n$ such that $h \notin \mathcal{L}_{n-1}$. For the target h that has been missed for a longer time and failed to be captured within a small survey area, e.g. for $h \in \mathcal{L}_n \bigcap \mathcal{L}_{n-1}$, a large covariance of $\tilde{p}_n(X_{n,h})$ should be set to allow for a more extensive search in a larger survey area. Finally, it is beneficial to incorporate the prior information of the sensor surveillance region in $\tilde{p}_n(X_{n,h})$. For example, $\tilde{p}_n(X_{n,h})$ should not have a high probability density over areas not covered by the sensor surveillance region or where the target is definitely not present.

6.4 Simulation

This section focuses on the evaluation of the proposed full VB-AbNHPP-RELO tracker in Algorithm 12 on multi-target tracking tasks in heavy clutter. Particularly, subsection 6.4.1 considers a tracking scenario under moderately heavy clutter where targets exhibit relatively more random movements, and subsection 6.4.2 presents a more challenging scenario with highly heavy clutter where targets first intersect in a small region and then disperse. Extensive experimental results are reported in both scenarios.

In subsections 6.4.1 and 6.4.2, we compare the proposed VB-AbNHPP-RELO with the standard VB-AbNHPP tracker (Algorithm 10 from Section 5.5) and other competing methods (specified later) to demonstrate its efficacy on the fixed number multi-target tracking tasks. Datasets are generated with various trajectories, target number, and measurement rates to simulate diverse tracking scenes with heavy clutter. Examples of single-time-step measurements in a challenging dense clutter scene are given in Fig. 6.5(c) and Fig. 6.8(c), where targets' measurements (red dots) are difficult to distinguish from the clutter (blue dots). Such an ambiguity may lead to multiple modes in the exact posterior of target states, whereas only one of them may capture the true targets' positions. An algorithm that fails to keep necessary modes can easily lose track of the targets, which explains why tracking in heavy clutter situations is challenging.

The methods evaluated in subsections 6.4.1 and 6.4.2 can be divided into two classes: The first class is methods based on sampling or maintaining multiple hypotheses, where several modes may be kept in the estimate per time step at the cost of computational time and memory. This class of methods includes Rao-Blackwellised Gibbs-AbNHPP (G-AbNHPP) (Algorithm 2, [50]) and the popular PMBM filter [154]. In addition,

besides the standard PMBM filter with a birth model (termed PMBM-B), we also implement an altered version of the PMBM filter with no birth and death models (PMBM-NB) as it matches our modelling assumptions and we found it can lead to better performance and faster implementation in the considered tasks. However, note that the original recycling step is kept in PMBM-NB, such that a birth process will still be carried out in a small region. Another tracking algorithm that belongs to this class is the Sum-Product-Algorithm-based tracker [156]. However, its computational time is much beyond the real-time processing requirement even for our simplest task ($K = 5$ in Section 6.4.1), under the recommended parameters in [156]. Therefore, we do not compare with this method in this chapter. The other class of evaluated algorithms in our experiments only keep one mode in the estimate per time step so that the tracking can be performed in the most efficient manner. These algorithms include the ET-JPDA filter in [147], our original VB-AbNHPP tracker and the proposed VB-AbNHPP-RELO, of which only the VB-AbNHPP-RELO has the capability to relocate the missed targets.

Some common settings and parameters for the simulations and tested algorithms are listed below:

Modelling assumptions in synthetic dataset We assume that targets move in a 2D surveillance area with each $X_{n,k} = [x^1_{n,k}, \dot{x}^1_{n,k}, x^2_{n,k}, \dot{x}^2_{n,k}]^T$, where $x^d_{n,k}$ and $\dot{x}^d_{n,k}$ $(d = 1, 2)$ indicate the k-th target's position and velocity in the d-th dimension, respectively. The following models and parameters are employed in all synthetic tracks generation and inference algorithms presented below. We utilise the constant velocity model with a transition density in (5.9), where we have $F_{n,k} = diag(F^1_{n,k}, F^2_{n,k})$, $Q_{n,k} = diag(Q^1_{n,k}, Q^2_{n,k})$, $B_{n,k} = 0$, with $F^d_{n,k}, Q^d_{n,k}$ $(d = 1, 2)$ being specified in (6.9).

$$F^d_{n,k} = \begin{bmatrix} 1 & \tau \\ 0 & 1 \end{bmatrix}, \quad Q^d_{n,k} = 25 \begin{bmatrix} \tau^3/3 & \tau^2/2 \\ \tau^2/2 & \tau \end{bmatrix}, \quad H^d = \begin{bmatrix} 1 & 0 \end{bmatrix}. \tag{6.9}$$

The total time steps are 50, and the time interval between measurements is $\tau = 1$s. The target is simulated as an extended target with an elliptical extent, and its covariance in (5.2) on page 143 is set to $R_k = 100\mathrm{I}_2$, $k = 1, ..., K$. Alternatively, R_k can also be interpreted as the covariance of the measurement noise. The measurement matrix in (5.2) is designed to generate the 2D positional measurements, i.e. $H = diag(H^1, H^2)$, where H^d $(d = 1, 2)$ is specified in (6.9).

Metric The metric utilised for measuring the tracking performance is the optimal sub pattern assignment (OSPA) [163]. The OSPA metric accounts for both state

estimation errors—differences between the estimated and true states of objects—and cardinality errors—differences in the number of objects between the estimation and the ground truth. Let $c > 0$, $1 \leq p < \infty$, $d(x, y)$ denote a metric for any $x, y \in \mathbb{R}^N$. Define $d^{(c)}(x, y) := \min\{d(x, y), c\}$ as the distance between x, y cut off at c. Let Π_n the set of permutations on $\{1, 2, \ldots, n\}$ for any $n \in \mathbb{N}$. Define $X = \{x_1, \ldots, x_m\}$ and $Y = \{y_1, \ldots, y_n\}$ as finite subsets of $\mathbb{R}^N$. For $m > n$, $d_p^{(c)}(X, Y) = d_p^{(c)}(Y, X)$. For $m \leq n$, the OSPA metric is defined as

$$d_p^{(c)}(X, Y) := \left(\frac{1}{n} \left[\min_{\pi \in \Pi_n} \sum_{i=1}^{m} \left(d^{(c)}(x_i, y_{\pi(i)}) \right)^p + c^p(n - m) \right] \right)^{1/p}$$

Here in this experiment, the order is set to $p = 1$ and the cut-off distance is $c = 50$.

Meanwhile, to evaluate the computational complexity, we monitor the CPU time (System: Intel(R) Core(TM) i9-9980 CPU@2.40GHz, 32 GB RAM) required at a single time step and averaged over all time steps. The overall CPU time presented in the table is the averaged value across all runs.

General parameter settings We set $I = 100, \epsilon = 0.01$ for both the standard VB-AbNHPP tracker and the VB-AbNHPP-RELO tracker. In these experiments, the proposed methods usually converge in fewer than 10 iterations under this setup.

For the VB-AbNHPP-RELO tracker, we search for the recently missed targets (for $h \in \mathcal{L}_n$ but not in $\mathcal{L}_{n-1}$) within a 490 radius, and all other missed targets (for $h \in \mathcal{L}_n \bigcap \mathcal{L}_{n-1}$) within a 1713 radius, both centred on the last tracked position. Such a circular area is modelled by a 95% confidence region of a $2D$ Gaussian with covariance $200\mathrm{I}_2$ and $700\mathrm{I}_2$ respectively. The Gaussian prior $\tilde{p}_n(X_{n,h})$ assigned for the missed target h is designed to cover the intersection of this circular region and the sensor surveillance area, that is, if this circular region is completely within the surveillance area, the $\tilde{p}_n(X_{n,h})$ has the corresponding mean and covariance $700\mathrm{I}_2$; otherwise it is adjusted to cover the intersection of these two regions. The velocities in $\tilde{p}_n(X_{n,h})$ are set with mean 0 and covariance 1600 for all dimensions since the targets are not biased towards any specific direction. According to Appendix 6.B.2, eligible initialisation threshold M_k^{init} is set to be equal to the relocation threshold M_k^{reloc}, and M_k^{reloc} is set with $P_{thres}^{reloc} = 0.5$. For the moderately heavy clutter scenes in subsection 6.4.1, the track loss detection parameters τ_k, M_k^{los} are set according to Appendix 6.B.1 with P_{thres}^{los} =7e-4, and the initialisation covariance C is $35^2\mathrm{I}_2$; for the highly heavy clutter scenarios in subsection 6.4.2, we set τ_k, M_k^{los} according to Appendix 6.B.1 with P_{thres}^{los} =5e-4 to achieve a slightly less sensitive track loss detection, and $C = 20^2\mathrm{I}_2$ to maintain a

reasonable relocation effectiveness with larger computational costs. See sections 6.2.2 and 6.2.3 for the rationale for selecting C.

For the G-AbNHNPP algorithm, the number of particles is set to 500 and the burn-in time is 100 iterations. We choose to compare with the fast Rao-Blackwellisation scheme of the G-AbNHPP since it outperforms other schemes in [50]. A higher tracking accuracy may be achieved with a larger particle size, but this comes with a longer computational time.

Since PMBM filters were originally devised for tracking a varying number of objects, we have heavily modified them for tracking a fixed number of K targets for a fair comparison in our experiment. In particular, for both PMBM-B and PMBM-NB, if their multi-Bernoulli global hypothesis with the highest weight includes more than K Bernoulli components, then the OSPA is evaluated with only the K components with the highest existence probability. Under this setup, we find that their cardinality estimates in our experiment are rarely wrong, and hence their OSPA merely reflects the localisation error in most cases. Moreover, both PMBM filters are set with the same accurate initialisation of target states, Poisson rate Λ, and target extent R as in other competing methods. Specifically, at time step 1, both PMBM filters start with a single multi-Bernoulli hypothesis with K components, each associated with an existence probability 1, and a ground truth target state, target extent, and Poisson rate (i.e. their priors all have an extremely small variance). The transitions of rates and shapes are removed such that the rates and shapes are nearly constant over time, allowing the PMBM filters to use ground truth information of Λ and R at every time step, like the other compared methods.

Other parameters of the PMBM-B and the PMBM-NB filters are listed as follows: For all scenarios, both the object's survival probability and the detection probability are set to 1. The ellipsoidal gate size in probability is 0.999. The number of global hypothesis is capped at 100. The DBSCAN algorithm is run with distance thresholds between 5 and 50, with a maximum number of 20 assignments. For PMBM-NB, the Poisson birth model is neglected by setting the intensity as a single component with weight 0.01 and Gaussian density with mean $[0, 0, 0, 0]^T$ and covariance $diag(0.01, 0.25, 0.01, 0.25)$, covering a negligible surveillance area. Note that there will still be Poisson intensity for undetected targets because of the recycling step in the PMBM algorithm. For PMBM-B, its Poisson intensity is set as a single component with weight 0.1 and Gaussian density with mean $[0, 0, 0, 0]^T$ and covariance $diag(40000, 16, 40000, 16)$, covering the circular region where the targets most likely appear/lose track. These parameters are manually tuned for good tracking performance and efficient implementation. For

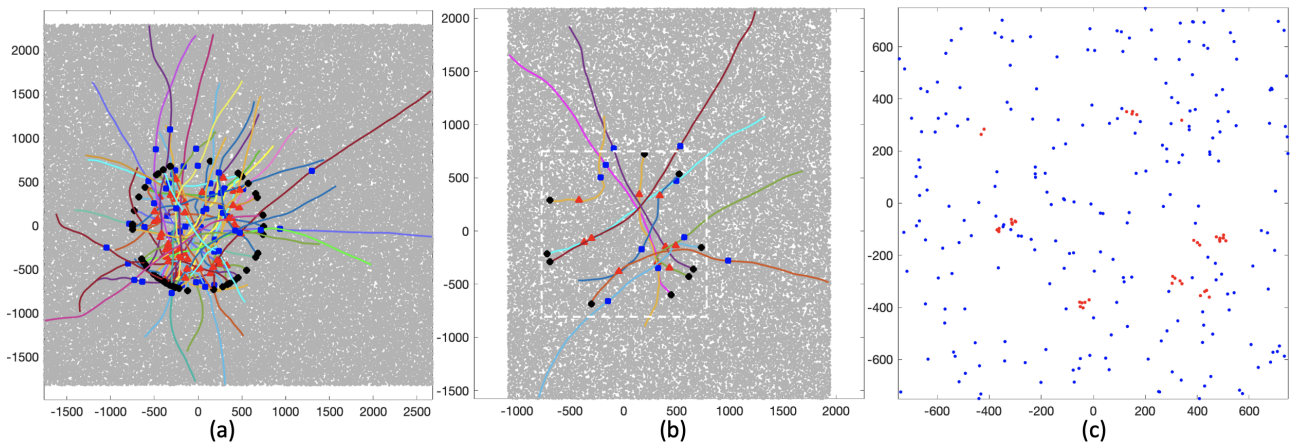


Fig. 6.5 Example synthetic trajectories and measurements in the tested dataset in Section 6.4.1. Target number K in figure (a) and (b) are 50 and 10 respectively. The black circles, red triangles and blue squares mark the targets' positions at the time step 1, 11 and 31. The dense grey dots in (a) and (b) are all measurements received from all time steps. Figure (c) shows the measurements received at the time step 11 from the same dataset as figure (b). Red dots are the measurements generated by the 10 targets (whose position is in the red triangles in figure (b)), and blue dots are the clutter received at the time step 11. Figure (c)'s range $[-750, 750] \times [-750, 750]$ corresponds to the white dashed square depicted in figure (b).

example, the grid for DBSCAN clustering is carefully selected such that a finer grid no longer improves the tracking accuracy but only introduces extra computational time in our tracking examples.

6.4.1 Tracking multiple targets under moderately heavy clutter

This section evaluates the proposed VB-AbNHPP trackers for tracking randomly moving targets that are closely spaced in moderately heavy clutter. Specifically, five scenarios are considered with the target number increasing from $5, 10, 20, 30$ to 50. For each scenario, 100 datasets are generated, each with a completely different set of synthetic targets' trajectories and measurements. Each synthetic trajectory in each dataset was randomly initiated on a circle with a radius of 750 from the origin $(0, 0)$, and it has an initial velocity of 30 pointing toward the origin. This simulates a scenario where targets move closer together before gradually spreading out. Two example tracks from 50 targets and 10 targets are shown in Fig. 6.5(a) and (b), where black circles, red triangles and blue squares mark the targets' positions at time step 1, 11 and 31 respectively. It can be seen that during the first 30 time steps, targets are

Table 6.1 Tracking performance comparisons for closely spaced randomly moving targets under moderately heavy clutter.

	OSPA (mean ±1 standard deviation) | CPU time (s)		
K	ET-JPDA	PMBM-NB	PMBM-B
5	7.24±2.31 | 0.02	6.19±1.32 | 0.44	7.29±2.26 | 4.36
10	21.42±4.13 | 0.08	6.65±1.23 | 1.86	8.09±2.11 | 5.63
20	22.47±2.73 | 0.22	8.67±2.29 | 2.43	12.17±2.94 | 11.57
30	24.09±2.26 | 0.39	10.20±1.80 | 3.24	15.63±2.74 | 16.63
50	25.06±1.82 | 0.86	14.09±1.71 | 6.14	19.97±2.32 | 40.98
	OSPA (mean ±1 standard deviation) | CPU time (s)		
K	G-AbNHPP	VB-AbNHPP	VB-AbNHPP-RELO
5	**5.65**±0.43 | 0.40	6.12±1.83 | 5e-5	5.70±0.41 | 3e-3
10	5.79±0.84 | 0.88	6.24±1.34 | 1e-4	**5.72**±0.35 | 0.01
20	6.16±0.95 | 1.98	7.05±1.34 | 3e-4	**5.95**±0.41 | 0.05
30	6.78±1.24 | 2.76	7.35±1.29 | 5e-4	**6.06**±0.33 | 0.14
50	7.95±1.46 | 4.12	8.64±1.13 | 1e-3	**6.45**±0.34 | 0.87

closely spaced, and track coalescence occurs frequently with its frequency and severity increasing with target number. After the 30-th time step, the targets gradually move away from each other and tracking becomes less difficult.

For all datasets of this moderately heavy clutter scene, we set the Poisson rate $\Lambda_k = 5$ for all targets ($k = 1, 2, ..., K$). The clutter density (i.e. Λ_0/V) is 10^{-4} per unit area, and correspondingly, the average Λ_0 for each scenario with target number K being $5, 10, 20, 30$ and 50 are $775, 1175, 1761, 1967$ and 2521, respectively. With such a high clutter, the measurements received from all time steps (showed as grey dots in Fig. 6.5(a) and (b)) are visually overlapped. The measurements received from a single time step are also plotted in Fig. 6.5(c) where the red dots are measurements generated by 10 targets that are located at the red triangles in Fig. 6.5(b), and the blue dots are the clutter received in this single time step. Such dense clutter causes ambiguity in targets' true positions, highlighting another challenge of our experiment, in addition to the frequent coalescences that occur when target numbers are high.

The tracking results for all methods are presented in Table 6.1. For each $K = 5, 10, ..., 50$, the mean OSPA and CPU time are first averaged over all 50 time steps, and then averaged over all 100 datasets. The standard deviation of these 100 average OSPA is also shown in Table 6.1. We can observe that our method has a promising performance in both time efficiency and tracking accuracy. In terms of efficiency, our

standard VB-AbNHPP tracker is the fastest algorithm, and its efficiency advantage becomes more evident as the target number increases. The proposed VB-AbNHPP-RELO tracker naturally requires extra computational time than the standard VB-AbNHPP tracker, but it is still more efficient than all other compared methods, and the advantages over the PMBM filter and G-AbNHPP are particularly significant. In terms of tracking accuracy, the proposed VB-AbNHPP-RELO outperforms all other compared methods in all considered scenarios, except for being comparable to G-AbNHPP with similar average OSPA values. Based on the experimental results, the remaining tested algorithms can be ranked in terms of their average OSPA from lowest to highest as follows: G-AbNHPP, PMBM-NB, PMBM-B, ET-JPDA, and our standard VB-AbNHPP tracker.

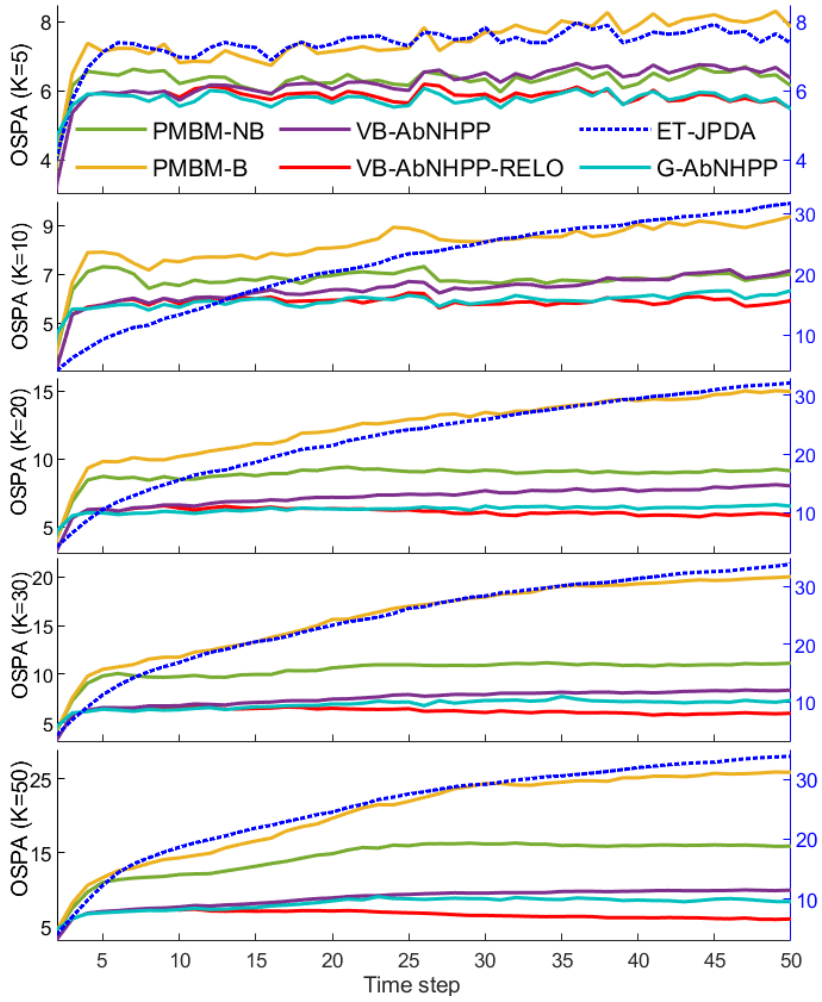


Fig. 6.6 Mean OSPA metric over 50 time steps for tracking scene in Section 6.4.1. Blue dashed line is associated with the right y-axis, and all other lines are with the left y-axis.

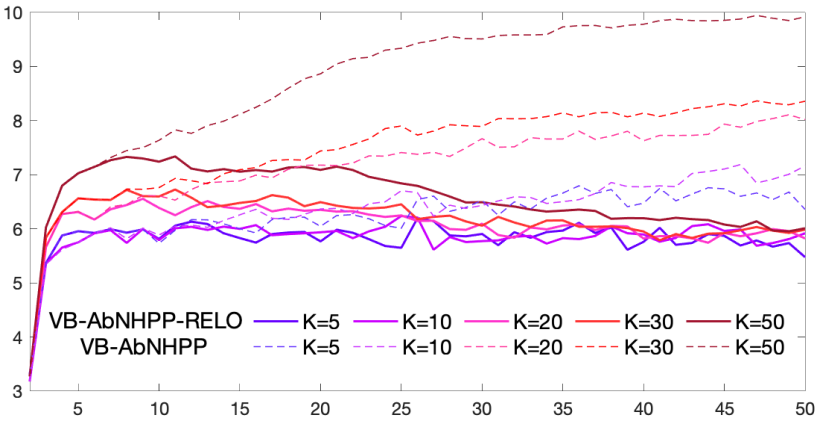


Fig. 6.7 Mean OSPA of VB-AbNHPP-RELO and standard VB-AbNHPP over 50 time steps for cases in Section 6.4.1.

To illustrate detailed tracking performance over time, the mean OSPA over 50 time steps for each scenario is plotted in Fig. 6.7. In the beginning stage of each tracking scene, the G-AbNHPP has the lowest mean OSPA, followed by our variational Bayes trackers. This advantage of G-AbNHPP lasts shorter as tracking scenarios become more challenging, i.e. when K grows. As time goes on, the proposed VB-AbNHPP-RELO outperforms G-AbNHPP and always achieves the lowest OSPA in the last few time steps, owing to the proposed effective relocation strategy. Such a phenomenon about G-AbNHPP and VB-AbNHPP-RELO is reasonable, and we analyse it below.

Recall that all tested methods are given an identical and accurate initialisation of target states, and the G-AbNHPP can retain different modes in the posterior with different samples at the cost of computational time and memory. In the beginning of each tracking scene, a limited number of samples in G-AbNHPP are likely to retain all necessary modes and thus all targets can be tracked successfully. In contrast, our efficient VB-AbNHPP-RELO always keeps a single mode at each time step, making it more likely to lose track. Although the proposed relocation strategy can retrieve the missed targets, the track loss prior to the successful relocation inevitably leads to a deterioration of the mean OSPA. Therefore, the G-AbNHPP typically outperforms the VB-AbNHPP-RELO in the beginning of a tracking task. However, in each new time step, there is always a chance that the limited number of samples in G-AbNHPP may miss some important mode that can potentially lead to track loss. Subsequently, as time goes on, G-AbNHPP is more likely to lose track and has an irreparably high OSPA due to its inability to relocate missed targets. For more challenging tracking tasks where there are more modes in the posterior, G-AbNHPP are less likely to retain all the necessary modes using the same sample size, resulting in more frequent track losses.

Accordingly, the OSPA difference (both in Table 6.1 and in the last few time steps in Fig. 6.7) between G-AbNHPP and VB-AbNHPP-RELO becomes more significant as K increases, since the latter maintains a stable tracking performance by constantly detecting and relocating the missed targets across all tracking scenes.

Although outperformed by our more efficient VB-AbNHPP-RELO in challenging tracking tasks (e.g. $K \geq 10$), the G-AbNHPP is still advantageous over all other tested methods. Other tested methods that retain multiple modes are PMBM-NB and PMBM-B, where different hypotheses of measurement associations are recorded at each time step. In particular, PMBM-NB does better than PMBM-B in all considered scenarios. However, they do not perform very well: apart from outperforming the ET-JPDA, both PMBM filters have a higher overall OSPA in Table 6.1 than all other methods, including our standard VB-AbNHPP that is also over thousands of times faster. This trend is also reflected in Fig. 6.7, where both PMBM filters always have a higher OSPA than the G-AbNHPP and our variational Bayes trackers, except that the PMBM-NB slightly outperforms our standard VB-AbNHPP tracker in the last few time steps in the cases $K = 5$ and $K = 10$. One factor that could lead to the high OSPA of PMBM filters is the poor performance of the clustering algorithm in this heavy clutter, which further results in inaccurate association hypotheses.

To present a clearer picture of the efficacy of the proposed missed target detection and relocation strategy, the mean OSPAs of the standard VB-AbNHPP tracker and the VB-AbNHPP-RELO are shown in Fig. 6.7. In the beginning of each tracking scene, the mean OSPAs of two trackers are overlapped, meaning that no track loss is detected in this period and both trackers perform exactly the same. As time goes on, while the OSPA of the standard VB-AbNHPP continues to grow, the OSPA of the VB-AbNHPP-RELO starts to decrease over the rest of time, validating clearly the superiority of the proposed detection and relocation strategy. In particular, the decline in the OSPA of the VB-AbNHPP-RELO is because 1) tracking tasks become easier in the later stage of our simulation as targets become progressively more separated; and 2) missed targets are successfully detected and relocated. Furthermore, we note that the OSPA of the VB-AbNHPP-RELO in Fig. 6.7 always decreases and finally stabilises at a similar value over different Ks, whereas the OSPA of the standard VB-AbNHPP and other methods in Fig. 6.6 increases and stabilises at a higher value as K increases. This demonstrates the robustness of our VB-AbNHPP-RELO tracker, i.e. despite the fact that the coalescence of more targets may result in a higher OSPA at some point, our effective relocation strategy can always retrieve missed targets and eventually reach a robust OSPA value when targets are separated from each other.

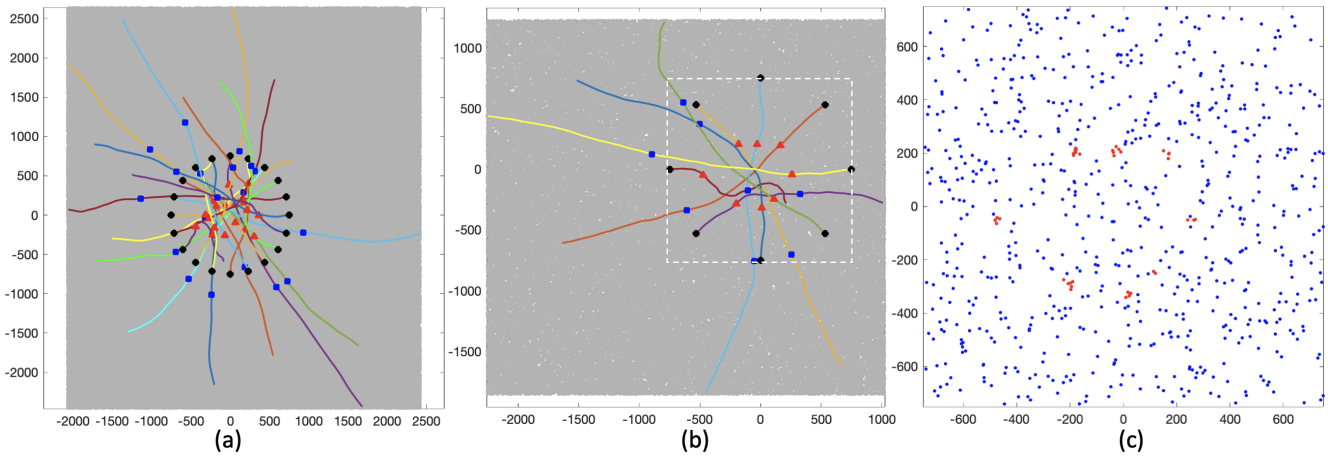


Fig. 6.8 Trajectories and example measurements in the two considered coalescence cases in Section 6.4.2. Target number K in figure (a) and (b) are 20 and 8 respectively. The black circles, red triangles and blue squares mark the targets' positions at time steps 1, 11 and 31. The dense grey dots in (a) and (b) are all measurements received from all time steps. Figure (c) shows the measurements received at the time step 11 from the same dataset as figure (b). Red dots are the measurements generated by the 10 targets (whose position is in the red triangles in figure (b)), and blue dots are the clutter received at the time step 11. Figure (c)'s range $[-750, 750] \times [-750, 750]$ corresponds to the white dashed square depicted in figure (b).

Finally, we note from Fig. 6.6 and 6.7 that the proposed VB-AbNHPP-RELO is the only tested method whose OSPA decreases after the first 10 time steps. This is again due to the significant advantage of our effective relocation method over other existing methods. On the contrary, the OSPAs of all other methods either grow or remain the same even when the coalescence is less severe, meaning that missed targets are seldom retrieved and more targets may be lost due to the high clutter. In particular, we may find that in such a heavy clutter tracking scene, the birth process in PMBM-B is unable to retrieve the missed targets as effectively as the proposed relocation strategy. This may be because: 1) there is a distinct mismatch between the birth process and our model assumptions, and 2) the birth process cannot cover the whole surveillance area due to the significant computational time required.

6.4.2 Analysis of two example coalescence scenarios under extremely heavy clutter

In this section, we consider two specific coalescence scenarios where multiple moving objects intersect around the origin point at a specific time. Targets' trajectories of

these two cases, which feature $K = 8$ and $K = 20$ objects, are shown in Fig. 6.8(b) and (a), respectively. In both scenarios, the initial positions of targets are equally spaced on a circle of radius 750 from the origin, and each target's initial velocity is 50 pointing towards the origin. Compared to the relatively random target's movement in Section 6.4.1, here the targets' trajectories are more straight, and all targets now intersect in a smaller region between time steps 11 and 25, see Fig. 6.8.

Here, two coalescence scenarios are designed under extremely heavy clutter to verify the advantages of our variational trackers in more challenging settings. In particular, the clutter density is 3×10^{-4} per unit area, which leads to a Poisson rate of 3038 for $K = 8$, and 6916 for $K = 20$. Poisson rates are 6 for all targets. For each scenario, 100 synthetic measurement sets are generated under ground truth trajectories in Fig. 6.8 on page 208, where grey dots denote one example measurement set for all time steps. Fig. 6.8(c) shows the measurements received at a single time step, where the true measurements (red dots) are largely buried in heavy clutter (blue dots), which can hardly be distinguished with human eyes. Note that Fig. 6.8(c) is on the same scale as Fig. 6.5(c), and thus we can easily visualise the increase in clutter density in Fig. 6.8(c), indicating an increased difficulty in performing tracking tasks.

Table 6.2 Tracking performance comparisons for intersecting targets under highly heavy clutter.

	OSPA (mean ±1 standard deviation) | CPU time (s)	
Method	8 targets	20 targets
ET-JPDA	22.03±2.48 | 0.16	35.40±1.17 | 0.96
PMBM-NB	7.53±2.14 | 2.15	9.29±1.94 | 6.68
PMBM-B	8.26±2.38 | 10.17	13.09±3.21 | 21.53
G-AbNHPP	6.10±1.42 | 1.31	6.48±1.28 | 4.07
VB-AbNHPP	6.36±1.78 | 2e-4	7.62±1.89 | 1e-3
VB-AbNHPP RELO	**5.63**±0.55 | 0.02	**6.16**±0.64 | 0.44

The overall tracking performance is presented in Table 6.2. It can be observed that the performance of all tested methods exhibits a pattern consistent with that in Table 6.1 in Section 6.4.1. For both two coalescence scenarios, the proposed VB-AbNHPP-RELO has the lowest mean OSPA and the second fastest implementation. Our standard VB-AbNHPP tracker is the fastest algorithm, but its tracking accuracy is surpassed by the G-AbNHPP and VB-AbNHPP-RELO. Note that in scenarios with 20 targets amidst heavy clutter, track losses become frequent, resulting in higher average OSPA values across all methods. This in turn showcases the advantage of

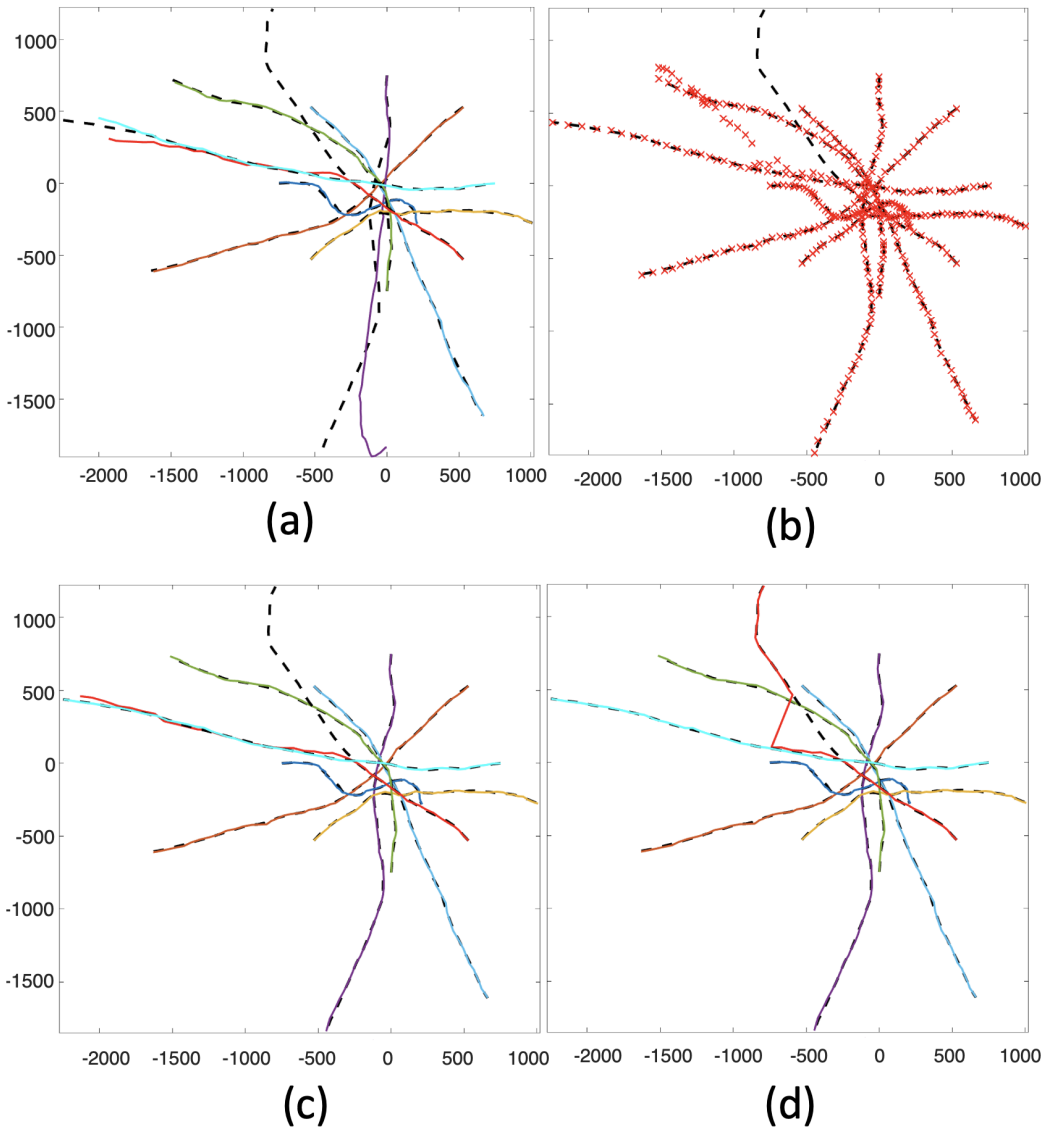


Fig. 6.9 Estimated trajectories (colored lines and red crosses) from (a) ET-JPDA, (b) PMBM-B, (c) VB-AbNHPP and (d) VB-AbNHPP RELO for a particular measurement set of $K = 8$ targets. The black dashed lines are the ground truth.

the proposed VB-AbNHPP-RELO tracker since it can relocate lost targets, resulting in better accuracy. However, the increased track loss events mean the VB-AbNHPP-RELO tracker resorts to the relocation strategy more often, and thus computational time rises, especially when contrasted with simpler situations with 8 targets. The ranking of all tested methods in terms of the mean OSPA (from low to high) for both scenarios is: VB-AbNHPP- RELO, G-AbNHPP, standard VB-AbNHPP, PMBM-NB, PMBM-B, and ET-JPDA.

We select a specific measurement set from 100 datasets for $K = 8$, where there are 'misleading' measurements that can trick trackers into steering the estimated trajectory of a particular target in the wrong direction. The estimated trajectories of four methods (ET-JPDA, PMBM-B, VB-AbNHPP and VB-AbNHPP RELO) are depicted in Fig. 6.9. For brevity, the trajectories of PMBM-NB and G-AbNHPP are not shown, but they are similar to Fig. 6.9(b) and (c) respectively. From Fig. 6.9 we can see that, once the target in red line intersects with the target in cyan line, all methods incorrectly track the red target, and the estimated tracks of the red and cyan targets coincide. However, the proposed VB-AbNHPP-RELO can detect track loss in a few time steps and successfully relocate it to its true position, whereas all other methods continue to track the red target falsely ever since its coalescence. In contrast, even though the track loss occur within the specified birth region in this example, the birth process in PMBM-B fails to localise the missed target. Consequently, VB-AbNHPP-RELO is the only method that correctly tracks all targets at the end of the task, demonstrating the superiority of our effective missed target detection and relocation strategy.

Finally, the mean OSPAs at 50 time steps for all methods are shown in Fig. 6.10 to reflect the tracking performance over time. All methods demonstrate similar patterns as in Fig. 6.7, and the analysis and comparisons of the general performance of these methods can be found in Section 6.4.1. Here we highlight two advantages of the proposed VB-AbNHPP-RELO: its robust tracking performance in handling the coalescence, and its lowest mean OSPA in the last few time steps. In particular, all other methods have a more evident increase in OSPA when the coalescence occurs between time steps 11 and 25, whilst the OSPA of VB-AbNHPP-RELO does not change greatly as missed targets are detected and relocated timely. For PMBM-NB and PMBM-B algorithms, the high OSPA at the moment of coalescence may be due to the limitation of the embedded clustering algorithm. We observe a drop (between time steps 17 and 19) from this temporarily high OSPA in PMBM filters when $K = 8$, but the drop does not happen in the more challenging case of $K = 20$. In the last few time steps in Fig. 6.8, the proposed VB-AbNHPP-RELO always has the lowest OSPA, demonstrating again the superiority of the proposed relocation strategy in the long-term tracking tasks.

6.5 Conclusion

In this chapter, we extend the standard VB-AbNHPP tracker in Section 5.5 from the previous chapter to a VB-AbNHPP-RELO tracker that can robustly track closely-

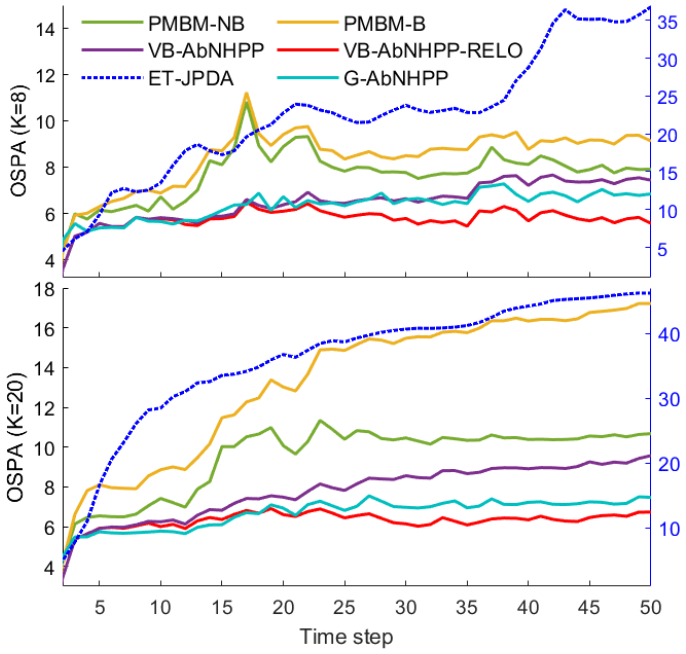


Fig. 6.10 Mean OSPA metric over 50 time steps for tracking scenes in Section 6.4.2. Blue dashed line is associated with the right y-axis, and all other lines are with the left y-axis.

spaced targets in heavy clutter by automatically detecting and relocating the missed targets. This is based on the proposed novel variational localisation strategy, which can efficiently localise the target from a large surveillance area in heavy clutter. The developed trackers offer fast and parallelisable implementations with superior tracking and estimation performance, making them promising candidates in large scale target tracking scenarios. In particular, when tracking a large number of objects under heavy clutter, the VB-AbNHPP-RELO demonstrates significantly better tracking performance over existing trackers in terms of both accuracy and efficiency.

The proposed variational localisation strategy may be extended to incorporate measurements from multiple time steps to offer a more reliable estimation. This localisation strategy can also facilitate joint detection and tracking task, and a more general VB-AbNHPP-RELO tracker that can handle other unknown variables, such as Poisson rates, measurement covariance, and the target number. This will be discussed in future work.

Appendix 6.A Further discussions on Remark 1 in Section 6.2.2

Although difficult to analyse theoretically, Remark 1 in Section 6.2.2 can be informally justified by looking into the iterative updates under a construction of the initialisation in (6.1). First, we describe some behaviours of these iterative updates, which will be used to illustrate Remark 1 in Section 6.2.2 in detail.

With the initial $q^{(0)}(\theta_n)$ in (6.1), only measurements that lie in the local region, i.e. the 95% confidence ellipse of $\mathcal{N}(m_s, C)$, have a considerable probability to associate with the target. Conditional on this $q^{(0)}(\theta_n)$, it then produces a pseudo measurement $\overline{Y}_n^{(0)}$ (evaluated according to (5.27)) that locates at the weighted average of those noticeable measurements with a covariance $\overline{R}_n^{(0)}$ (i.e. evaluated according to (5.26)). For a local region that contains at least one measurement, the denominator in (5.26) is considerable, leading to a much smaller covariance $\overline{R}_n^{(0)}$ compared to the large positional covariance of the uncertain prior $p(X_{n,1})$. Consequently, according to (5.25), the position in the updated $q^{(1)}(X_{n,1})$ at the first iteration has a mean that is approximately equal to $\overline{Y}_n^{(0)}$, and a covariance close to $\overline{R}_n^{(0)}$. Empirically, the next updated $q^{(2)}(X_{n,1})$ will typically cover the measurement nearest to $\overline{Y}_n^{(0)}$. Then,

1) If there are dense measurements near or within the high confidence region of $q^{(2)}(X_{n,1})$: The subsequent iterative coordinate ascent updates will refine the $q(X_{n,1})$ at a smaller scale and converge to the $q^*(X_{n,1})$ that encompasses those dense measurements in the end. However,

2) If measurements are distributed more evenly or scattered around the high confidence region of $q^{(2)}(X_{n,1})$: The subsequent iterative coordinate ascent updates will enlarge the covariance of $q(X_{n,1})$ to cover as many of those measurements as possible, and finally converge to the $q^*(X_{n,1})$ that equals the uncertain prior $p(X_{n,1})$, i.e. $q^*(X_{n,1}) = p(X_{n,1})$.

Remark 1 in Section 6.2.2 holds for a properly chosen C since in this case, we expect $\overline{Y}_n^{(0)}$, which is approximately the weighted average of the measurements covered by the 95% confidence ellipse of $\mathcal{N}(m_s, C)$, is close to the location with the greatest density of measurements in this local area. Then the converged $q^*(X_{n,1})$ is very likely to capture this most probable target location. However, if C is too large, Remark 1 in Section 6.2.2 hardly stands. Since in this case, the 95% confidence ellipse of $\mathcal{N}(m_s, C)$ $(s = 1, 2, ..., N)$ will include too many uniformly distributed clutter, resulting in their weighted average being close to the center of this local area, i.e. $\overline{Y}_n^{(0)} \approx m_s$. Subsequently, depending on how the measurement(s) within the high confidence region of $q^{(2)}(X_{n,1})$ are distributed,

the CAVI either converges to $q^*(X_{n,1})$ that encompasses the dense measurements nearest to m_s, or traps in the local optimum with $q^*(X_{n,1}) = p(X_{n,1})$.

Appendix 6.B Parameter selection for missed objects detection and relocation

In this appendix, we discuss the choice of the track loss detection parameters τ_k, M_k^{los}, relocation threshold M_k^{reloc}, and eligible initialisation threshold M_k^{init} $(k = 1, 2, ..., K)$ for the developed full VB-AbNHPP-RELO tracker. We assume the target and clutter rate Λ and measurement covariance R are known to us. We will show that τ_k, M_k^{los}, M_k^{reloc}, M_k^{init} can be automatically selected by specifying probabilities $P_{thres}^{los} \in (0, 1)$ and $P_{thres}^{reloc} \in (0, 1)$ (both defined below) respectively.

6.B.1 Guide for the choice of τ_k and M_k^{los}

When the event E in (6.7) on page 196 is used as the track loss criterion, the higher the M_k^{los}, the less likely that event E will happen, and the more sensitive the algorithm is to the potential track loss. A lower τ_k leads to an event E that considers the $\hat{M}_{n,k}$ from more recent time steps, and hence better reflecting the timely tracking performance. However, if τ_k and Λ_k are both very low (e.g. $\tau_k = 1$ and $\Lambda_k = 1$), it is very likely that no measurement is generated from the target k during these τ_k time steps, which makes the event E likely to occur even for a successful tracker. Therefore, a trade-off has to be made when choosing τ_k and M_k^{los}.

This subsection describes a strategy to automatically select suitable τ_k and M_k^{los} such that the event E in (6.7) rarely happens for a successful tracker. We define $P_{thres}^{los} \in (0, 1)$ as the probability of the rare event E occurring, given that target k has been successfully tracked. Subsequently, the rationale behind our track loss criterion can be explained with the idea of hypothesis testing. Specifically, if such a rare event E (in terms of a successful tracker) occurs, it implies that the target may not be properly tracked. Furthermore, since the event E is more likely to happen when the algorithm loses track of the target (as discussed in Section 6.3.1), we conclude that the track of the target k has been lost.

To select τ_k and M_k^{los}, we first specify the value of P_{thres}^{los}. Then for each $k = 1, 2, ..., K$, τ_k and M_k^{los} are evaluated based on the specified P_{thres}^{los} according to three steps:

Step 1: Set $\tau_k = \left\lceil \frac{1}{\Lambda_k} \log \frac{1}{P^{thres}} \right\rceil$, where $\lceil \cdot \rceil$ is the ceil function.

Step 2: Obtain a continuous increasing function $\tilde{F}_{\tau_k\Lambda_k} : [0,\infty) \to [F_{\tau_k\Lambda_k}(0),\infty)$ by interpolating (e.g. Spline interpolation) $(x, \tilde{F}_{\tau_k\Lambda_k}(x))$ from the following series of coordinates: $(0, F_{\tau_k\Lambda_k}(0))$, $(1, F_{\tau_k\Lambda_k}(1))$, $(2, F_{\tau_k\Lambda_k}(2)), ...$, where $F_{\tau_k\Lambda_k}$ is the exact Poisson cumulative distribution function (CDF) with rate parameter $\tau_k\Lambda_k$.

Step 3: Find the M_k^{los} such that $\tilde{F}_{\tau_k\Lambda_k}(M_k^{los}) = P_{thres}^{los}$.

We now explain the rationale behind these steps. Note that the exact CDF for the decimal variable $\sum_{t=n-\tau_k+1}^{n} \hat{M}_{t,k}$ (i.e. the estimated cumulative measurement numbers in (6.7)) is intractable. However, according to the additive property of Poisson distribution, the integer variable $\sum_{t=n-\tau_k+1}^{n} M_{t,k}$ (i.e. the exact cumulative measurement numbers) has a Poisson CDF $F_{\tau_k\Lambda_k}$. With the intuition that a successful tracker should not produce an estimate $\hat{M}_{t,k}$ that greatly deviates from the exact $M_{t,k}$, we use $\tilde{F}_{\tau_k\Lambda_k}$, i.e. a continuous increasing approximation of $F_{\tau_k\Lambda_k}$ constructed in step 2, as an approximate CDF for the estimate $\sum_{t=n-\tau_k+1}^{n} \hat{M}_{t,k}$ generated from a successful tracker. A noteworthy property of $\tilde{F}_{\tau_k\Lambda_k}$ is that it is continuous increasing, which makes it better to describe the decimal variable $\sum_{t=n-\tau_k+1}^{n} \hat{M}_{t,k}$ than the step-wise function $F_{\tau_k\Lambda_k}$.

Subsequently, the probability of the event E in (6.7) conditional on the target k having been successfully tracked can be approximated by $\tilde{F}_{\tau_k\Lambda_k}(M_k^{los})$, i.e. $P_{thres}^{los} \approx \tilde{F}_{\tau_k\Lambda_k}(M_k^{los})$. Therefore, once τ_k and P_{thres}^{los} are specified, we can find M_k^{los} by Step 3. Note that the uniqueness of M_k^{los} found in Step 3 is guaranteed by the increasing property of $\tilde{F}_{\tau_k\Lambda_k}$, and the existence of M_k^{los} is guaranteed by the continuous property of $\tilde{F}_{\tau_k\Lambda_k}$ and the fact that $\tilde{F}_{\tau_k\Lambda_k}(0) = F_{\tau_k\Lambda_k}(0) \leq P_{thres}^{los}$. This inequality is always satisfied owing to τ_k constructed in Step 1. Specifically, τ_k in Step 1 is essentially the minimal integer that satisfies $F_{\tau_k\Lambda_k}(0) \leq P_{thres}^{los}$. Such a small τ_k ensures our track loss criterion E reflects timely tracking performance.

6.B.2 Guide for the choice of M_k^{reloc} and M_k^{init}

It is practically useful to choose M_k^{reloc} based on a specified probability $P_{thres}^{reloc} \in (0,1)$, where P_{thres}^{reloc} is the probability of the effective relocation criterion in (6.8) being satisfied, conditional on a successful relocation. On the one hand, we typically require $P_{thres}^{reloc} > 0.2$ to ensure the effective relocation criterion is easy to meet, otherwise the algorithm may take too many steps to find a convincing position for missed target. On the other hand, we also need to ensure P_{thres}^{reloc} is not too high, since a high P_{thres}^{reloc} often leads to frequent relocations and bears a high chance of a false relocation that may deteriorate the tracking performance.

The relocation threshold M_k^{reloc} can be selected according to P_{thres}^{reloc} in a similar fashion as in Appendix 6.B.1. Specifically, we first use a continuous increasing function $\tilde{F}_{\Lambda_k} : [0, \infty) \to [F_{\Lambda_k}(0), \infty)$ to approximate the CDF for the variable $\sum_{j=0}^{M_n} q_n^w(\theta_{n,j} = k)$ in (6.8) conditional on a successful relocation with the s-th initialisation. This approximate CDF $\tilde{F}_{\Lambda_k}$ can be obtained by carrying out Step 2 described in Appendix 6.B.1 with τ_k being 1. Then find M_k^{reloc} such that $\tilde{F}_{\Lambda_k}(M_k^{reloc}) = 1 - P_{thres}^{reloc}$.

Finally, we discuss the choice of the eligible initialisation threshold M_k^{init}, which is employed in Algorithm 11 to skip unnecessary initiasations to accelerate the relocation procedure. This eligible initialisation threshold M_k^{init} can simply be set as $M_k^{reloc} - 1$ or $M_k^{reloc} - 2$. Such a design of M_k^{init} satisfies $M_k^{init} < M_k^{reloc}$. Therefore the algorithm never skip exploring any local region that encompasses enough measurements to meet the effective relocation criterion in (6.8). Moreover, by setting a lower M_k^{init} (e.g. $M_k^{init} = M_k^{reloc} - 2$), the algorithm also explores local regions that include fewer measurements, which sometimes is necessary. For example, when the true missed target is at the edge of a local region, only a few target-oriented measurements may lie in this local region. In this case, there may be less than M_k^{init} measurements in this local region. However, with the corresponding initialisation, it is possible for CAVI to localise the missed target and the effective relocation criterion is still met.

Appendix 6.C Derivation of the ELBO for the relocation strategy

In this appendix, we derive the explicit form of the ELBO in (6.3) up to an additive constant for implementing the relocation strategy. In particular, we compute $\mathcal{F}_{n,h}(q_n(\theta_n), q_n(X_{n,h})) - c$, where c is the constant from (6.3). This can be expressed as follows, based on (6.3):

$$\begin{aligned}\mathcal{F}_{n,h}(q_n(\theta_n), q_n(X_{n,h})) - c = &- \mathrm{KL}(q_n(\theta_n)||p(\theta_n|M_n, \Lambda)) - \mathrm{KL}(q_n(X_{n,h})||\tilde{p}_n(X_{n,h})) \\ &+ \mathrm{E}_{q_n(\theta_n)q_n(X_{n,h})q_n^*(X_{n,h-})} \log p(Y_n|\theta_n, X_n). \end{aligned} \tag{6.10}$$

The first negative KL divergence is

$$\begin{aligned}-\mathrm{KL}(q_n(\theta_n)||p(\theta_n|M_n, \Lambda)) &= \sum_{j=1}^{M_n} \sum_{k=0}^{K} q_n(\theta_{n,j} = k) \log \frac{p(\theta_{n,j} = k|M_n, \Lambda)}{q_n(\theta_{n,j} = k)} \\ &= \sum_{j=1}^{M_n} \sum_{k=0}^{K} q_n(\theta_{n,j} = k) \log \frac{\Lambda_k}{\Lambda_{\mathrm{sum}} q_n(\theta_{n,j} = k)}. \end{aligned} \tag{6.11}$$

Denote the mean and covariance of $\tilde{p}_n(X_{n,h})$ as $\tilde{\mu}_{n,h}$ and $\tilde{\Sigma}_{n,h}$; and recall that $q_n(X_{n,h}) = \mathcal{N}(X_{n,h}; \mu^h_{n|n}, \Sigma^h_{n|n})$.Then the second negative KL divergence between two multivariate normal in (6.10) is

$$\begin{aligned}
&- \mathrm{KL}(q_n(X_{n,h})||\tilde{p}_n(X_{n,h})) \\
&= -\frac{1}{2}\left[\mathrm{Tr}(\tilde{\Sigma}^{-1}_{n,h}\Sigma^h_{n|n}) + (\tilde{\mu}_{n,h} - \mu^h_{n|n})^\top \tilde{\Sigma}^{-1}_{n,h}(\tilde{\mu}_{n,h} - \mu^h_{n|n}) - D_x + \log\frac{\det\tilde{\Sigma}_{n,h}}{\det\Sigma^h_{n|n}}\right], \quad (6.12)
\end{aligned}$$

where D_x is the dimension of $X_{n,h}$. Note evaluating this divergence requires inverting $\tilde{\Sigma}_{n,h}$. Since $\tilde{\Sigma}_{n,h}$ is an uninformative prior assigned for the missed target, we can always let it be a diagonal matrix, which leads to a simple and efficient inversion.

By using the formula for the expectation of quadratic forms, the last term in (6.10) is

$$\begin{aligned}
&\mathrm{E}_{q_n(\theta_n)q_n(X_{n,h})q^*_n(X_{n,h-})} \log p(Y_n|\theta_n, X_n) \\
&= \sum_{j=1}^{M_n} q_n(\theta_{n,j}=0)\log\frac{1}{V} + \sum_{j=1}^{M_n}\sum_{k=0}^{K} q_n(\theta_{n,j}=k)\left[-\frac{D}{2}\log 2\pi - \frac{1}{2}\log\det R_k\right] \\
&\quad + \sum_{j=1}^{M_n}\sum_{k=0}^{K} q_n(\theta_{n,j}=k)\left[-\frac{1}{2}\mathrm{E}_{q_n(X_{n,h})q^*_n(X_{n,h-})}(Y_{n,j} - HX_{n,k})^\top R_k^{-1}(Y_{n,j} - HX_{n,k})\right] \\
&= -\frac{1}{2}\sum_{j=1}^{M_n}\sum_{k=0,k\neq h}^{K} q_n(\theta_{n,j}=k)U^{k*}_n \\
&-\frac{1}{2}\sum_{j=1}^{M_n} q_n(\theta_{n,j}=h)\left[(Y_{n,j} - H\mu^h_{n|n})^\top R_k^{-1}(Y_{n,j} - H\mu^h_{n|n}) + \mathrm{Tr}(R_k^{-1}H\Sigma^h_{n|n}H^\top) + \log\det R_k\right] \\
&+ \left(\frac{D}{2}\log 2\pi + \log\frac{1}{V}\right)\sum_{j=1}^{M_n} q_n(\theta_{n,j}=0) - \frac{DM_n}{2}\log 2\pi \quad (6.13)
\end{aligned}$$

where we use (5.68) to obtain the last line, and U^{k*}_n is defined as

$$U^{k*}_n = (Y_{n,j} - H\mu^{k*}_{n|n})^\top R_k^{-1}(Y_{n,j} - H\mu^{k*}_{n|n}) + \mathrm{Tr}(R_k^{-1}H\Sigma^{k*}_{n|n}H^\top) + \log\det R_k. \quad (6.14)$$

Finally the ELBO (with an additive constant) in (6.10) is the summation of (6.11), (6.12), and (6.13), i.e.

$$
\begin{aligned}
&\mathcal{F}_{n,h}(q_n(\theta_n), q_n(X_{n,h})) - c \\
&= \sum_{j=1}^{M_n} \sum_{k=0}^{K} q_n(\theta_{n,j} = k) \log \frac{\Lambda_k}{\Lambda_{\text{sum}} q_n(\theta_{n,j} = k)} - \frac{1}{2} \sum_{j=1}^{M_n} \sum_{k=0, k\neq h}^{K} q_n(\theta_{n,j} = k) U_n^{k*} \\
&- \frac{1}{2} \sum_{j=1}^{M_n} q_n(\theta_{n,j} = h) \left[(Y_{n,j} - H\mu_{n|n}^h)^\top R_k^{-1} (Y_{n,j} - H\mu_{n|n}^h) + \text{Tr}(R_k^{-1} H \Sigma_{n|n}^h H^\top) + \log\det R_k \right] \\
&- \frac{1}{2} \left[\text{Tr}(\tilde{\Sigma}_{n,h}^{-1} \Sigma_{n|n}^h) + (\tilde{\mu}_{n,h} - \mu_{n|n}^h)^\top \tilde{\Sigma}_{n,h}^{-1} (\tilde{\mu}_{n,h} - \mu_{n|n}^h) + \log \frac{\det \tilde{\Sigma}_{n,h}}{\det \Sigma_{n|n}^h} \right] \\
&+ \left(\frac{D}{2} \log 2\pi + \log \frac{1}{V} \right) \sum_{j=1}^{M_n} q_n(\theta_{n,j} = 0) + \frac{D_x}{2} - \frac{DM_n}{2} \log 2\pi,
\end{aligned} \tag{6.15}
$$

where U_n^{k*} is given in (6.14), and the last two terms in the final line are constants that can be omitted when computing the ELBO. Additionally, $U_n^{k*}(k = 1, 2, ..., K)$ is also a constant when implementing the relocation strategy, and it can be precomputed using the output from the standard tracker.

Chapter 7

Conclusions and Future Work

The effective implementation of Bayesian inference for time series processing tasks, such as object tracking, relies on the careful development and integration of both stochastic modelling and approximate Bayesian inference, as demonstrated in this thesis. This thesis makes novel contributions in both modelling and inference, with the application aspects focusing on object tracking and intent inference. We now summarise the contribution and outline future work in terms of the methodology and applications aspects, respectively.

7.1 Conclusions

From the theoretical and methodology aspects, Chapters 2 and 3 emphasise the development of innovative stochastic models using stochastic differential equation, while Chapters 4, 5 and 6 mainly contribute novel approximate Bayesian inference (particularly variational Bayes) schemes.

Stochastic modelling The notable contributions in stochastic modelling include:

- A novel stable Lévy modelling framework is introduced in Chapter 3 for tracking applications. This framework improves conventional continuous-time dynamic models by replacing the Gaussian driven noise with its natural heavy-tailed extension, the α-stable noise, as suggested by the generalised central limit theorem. More likely to exhibit extreme values, the resulting stable Lévy state-space models, expressed as continuous-time Lévy processes, can better capture sharp changes in the state, showcasing their capability for modelling erratic maneuvering behaviors, such as swift turns or abrupt accelerations. The proposed models are

constructed in a conditionally Gaussian form, thus ensuring the tractability of employing Rao-Blackwellisation for enhanced Monte Carlo inference. Improved tracking performance is demonstrated with maritime vessel data compared to a conventional Gaussian dynamic model.

- Introduction and exploration of various stochastic models exhibiting mean-reverting behavior for modelling motion towards a specified destination, as demonstrated in Chapters 2 and 3 for intent inference applications. These models cover a range of continuous time stochastic processes, including a class of multidimensional Ornstein-Uhlenbeck (OU) processes, jump diffusions, stable Lévy processes, and bridging distributions or conditioned Markov processes (by conditioning the future arriving state).

Approximate Bayesian inference The notable contributions in approximate Bayesian inference include:

- Chapter 4 proposes a Conditionally Factorised Variational Bayes (CVB) algorithm to address the challenges of variational Bayes for highly correlated variables, where standard mean-field approximations lead to large inference errors. The algorithm is based on a proposed novel conditionally factorised variational family, which can account for variable dependencies with user-selected details and encompasses the standard mean-field family as a special case. With the derived coordinate ascent updates, CVB ensures a better Evidence Lower Bound (ELBO) with a finer conditional structure, thus offering a flexible trade-off between computational cost and inference accuracy. Additionally, an importance sampling-based CVB algorithm is developed for approximating intractable updates, yielding an estimated ELBO as a byproduct. We prove useful properties of the algorithm, such as the monotonically increasing estimated ELBO, discuss its applicability, and showcase its guaranteed performance improvements over standard mean-field coordinate ascent variational inference (CAVI).

- Chapter 6 introduces an innovative approach to seek the global optimum in variational Bayes for the heavy clutter localisation problem by employing multiple independent (hence parallelisable) CAVIs. Each CAVI features a specifically designed initialisation to target a selected small region, enabling it to find the optimal solution within that region. By comparing the resulting ELBOs, we identify the best local optima and enhance global optimum discovery.

- A unified coordinate ascent variational filtering framework that allows for both filtering with and without static parameter learning is presented in Section 5.3. The framework is designed for a generic dynamic system, enabling us to flexibly treat some newly received information (e.g. the cardinality of measurements) as a known quantity for both prediction and update steps. Additionally, the framework can handle likelihood functions that are known only up to a normalisation constant and can even accommodate known functions related to the state variable, rather than directly involving the measurement.

Multi-object tracking and intent inference From the application aspects, the intent inference and single object tracking are considered in Chapters 2 and 3, while Chapters 5 and 6 focus on multi-object tracking tasks. Our notable contributions for multi-object tracking and intent inference can be summarised as:

- Chapter 5 introduces the variational Bayes association-based NHPP tracker (VB-AbNHPP) tracker, which efficiently performs tracking, data association, and learning of target and clutter rates with a parallelisable implementation. The tracker can also be extended for online learning of other static parameters, such as object extent. The proposed tracker greatly outperforms competing methods in terms of efficiency. In the simulated relatively simple tracking scenes we have tested, its performance is comparable to that of a Markov chain Monte Carlo (MCMC)-based tracker, while still outperforming other methods, showcasing its effectiveness in these scenarios.

- Chapter 6 addresses highly challenging tracking scenarios involving closely-spaced objects and extremely heavy clutter by extending the VB-AbNHPP tracker. A novel variational localisation strategy that enables the rapid rediscovery of missed targets in extensive surveillance areas is introduced. Additionally, a novel track loss detection strategy for the VB-AbNHPP tracker is developed. By integrating these strategies into the standard VB-AbNHPP tracker, a robust VB-AbNHPP tracker capable of detecting and recovering from track loss is created. This enhanced tracker significantly outperforms existing trackers in terms of both accuracy and efficiency in simulated challenging environments.

- To tackle the intent inference task, this thesis models the intended destination as an unknown parameter that drives the dynamic model. In Chapter 2, a Bayesian intent inference framework is presented for predicting the destination from a set of fixed and known endpoints. Then, Chapter 3 introduces an efficient

method for predicting a static or dynamically varying intended destination, which can be any spatial point or region within the surveillance area. The developed methodologies have demonstrated effectiveness in handling both scenarios where the tracked object is moving in a static environment and when experiencing significant external perturbations. In Section 2.5 and 3.6, their performance has been validated using real datasets involving predictive touch applications with 3D freehand pointing gestures, as well as 2D pointing-click tasks with a mouse cursor.

7.2 Future Work

The research presented in this thesis lays a solid foundation for advanced studies in stochastic modelling, approximate Bayesian inference, and applications in multi-object tracking and intent inference. Several potential directions for future work are discussed as follows.

For the stochastic modelling, there are a number of extensions based on the developments in Chapter 2 and 3. Firstly, for scenarios where knowledge of the directional distribution of external perturbations is available, non-isotropic stable Lévy state space models can be constructed as in [21]. Additionally, other Lévy processes with less heavy-tailed driven noise can be developed for tracking less severely manoeuvring objects. To generate such noises, the driving process could be based on generic hyperbolic processes instead of α-stable processes. See [19] for the formulation and simulation of these processes, which can still yield conditionally Gaussian transition densities by sampling the subordinator generalised inverse Gaussian processes. Regarding the development of stochastic processes for modelling the destination reverting behaviour, future work could involve introducing mean-reversion terms in non-linear/non-Gaussian dynamic models that have been found useful for certain types of motions. For instance, mean-reversion effects are introduced in [92, 164] for the intrinsic model, which can effectively characterise both curvilinear motion and linear acceleration. Furthermore, modelling and inferring intent in group tracking scenarios [142, 165], where targets exhibit interactions with each other while still moving towards their intended destinations, can be another promising research direction. This approach could help improve the accuracy and effectiveness of tracking and inference tasks in complex, multi-object situations where individual and group dynamics coexist.

With regards to approximate Bayesian inference methodology, the CVB algorithm from Chapter 4 forms the foundation for a sequential implementation featuring the

importance sampling particle filter. Future work could expand this implementation into a filtering and parameter learning framework, following a similar approach to the one presented in Section 5.3. This extension would further improve the versatility and applicability of the CVB algorithm in a variety of sequential inference tasks. Future work on global optimum searching in variational Bayes for localisation problems can be directly extended to include measurements from multiple time frames. By incorporating additional information about the object's trajectory over time, this approach would naturally lead to enhanced localisation results and may even allow for the reliable inference of other unknown variables, such as Poisson rates and measurement covariance, albeit at a slightly higher computational cost.

In terms of the multi-object tracking, it is valuable to expand the proposed variational localisation strategy to support joint detection and tracking tasks (similar to those presented in [166, 167]), as well as to develop a more general VB-AbNHPP tracker with relocation strategy (VB-AbNHPP-RELO) tracker that can handle other unknown variables such as Poisson rates, measurement covariance, and target numbers. This would enhance its versatility in tracking varying numbers of objects. Moreover, the aforementioned extension of the sequential implementation of CVB could be utilised to maintain multiple modes by retaining a small number of samples, which is anticipated to improve tracking performance while still requiring less computational power compared to MCMC-based trackers in [166]. Additionally, it is viable to develop a decentralised implementation of the VB-AbNHPP tracker, which is expected to achieve equivalent performance to the centralised implementation. This makes it a promising candidate for tracking within sensor networks, as compared to other sub-optimal sensor fusion methods (e.g. [168]). The preliminary results of these developments are showcased in [169] and will be further explored in subsequent research.

As for the intent inference, we first note that the future work mentioned earlier on constructing mean-reverting processes can lead to new intent inference applications for various tracking scenarios. Furthermore, treating intent as a latent variable and estimating it along with the kinematic state, as introduced in Chapter 3, provides an efficient method for predicting static or dynamically varying intent. See also for its application in detecting malicious intent, such as the intent to enter a restricted area, as demonstrated in [170]. The efficiency of this method makes it a promising candidate for handling more complex intent or for integration with complicated non-linear dynamic models, where state estimation using approximate inference is already computationally demanding. Lastly, such an intent inference framework can be easily incorporated into

the developed VB-AbNHPP tracker, enabling the efficient inference of both intent and kinematics states of the tracked targets under cluttered conditions.

References

[1] A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin, *Bayesian data analysis*. Chapman and Hall/CRC, 1995.

[2] M. Bayes and M. Price, "An essay towards solving a problem in the doctrine of chances. By the late Rev. Mr. Bayes, F. R. S. communicated by Mr. Price, in a letter to John Canton, A. M. F. R. S." *Philosophical Transactions (1683-1775)*, vol. 53, pp. 370–418, 1763. [Online]. Available: http://www.jstor.org/stable/105741

[3] P. S. de Laplace, "Memoir on the probability of the causes of events," *Statistical Science*, vol. 1, no. 3, pp. 364–378, 1986, translated from the original French by S. M. Stigler. Originally published as "Mémoire sur la probabilité des causes par les évènements" (1774).

[4] Y. Bar-Shalom and X.-R. Li, *Multitarget-multisensor tracking: principles and techniques*. YBs Storrs, CT, 1995, vol. 19.

[5] Y. Bar-Shalom, P. Willett, and X. Tian, *Tracking and Data Fusion: A Handbook of Algorithms*. YBS Publishing, 2011.

[6] M. Fanaswala and V. Krishnamurthy, "Spatiotemporal trajectory models for metalevel target tracking," *IEEE Aerospace and Electronic Systems Magazine*, vol. 30, no. 1, pp. 16–31, 2015.

[7] B. I. Ahmad, J. K. Murphy, S. Godsill, P. Langdon, and R. Hardy, "Intelligent interactive displays in vehicles with intent prediction: A Bayesian framework," *IEEE Signal Processing Magazine*, vol. 34, no. 2, pp. 82–94, 2017.

[8] R. F. Bass, *Stochastic processes*. Cambridge University Press, 2011, vol. 33.

[9] S. M. Iacus *et al.*, *Simulation and inference for stochastic differential equations: with R examples*. Springer, 2008, vol. 486.

[10] X. R. Li and V. P. Jilkov, "Survey of maneuvering target tracking. Part I. Dynamic models," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 39, no. 4, pp. 1333–1364, 2003.

[11] P. Alquier, "Approximate Bayesian inference," *Entropy*, vol. 22, no. 11, p. 1272, 2020.

[12] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer, 2006.

[13] B. Oksendal, *Stochastic differential equations: an introduction with applications*. Springer Science & Business Media, 2013.

[14] S. Särkkä and A.Solin, *Applied Stochastic Differential Equations*, ser. Institute of Mathematical Statistics Textbooks. Cambridge University Press, 2019.

[15] F. C. Klebaner, *Introduction to stochastic calculus with applications*. World Scientific Publishing Company, 2012.

[16] A. BESKOS and G. O. ROBERTS, "Exact simulation of diffusions," *The Annals of Applied Probability*, vol. 15, no. 4, pp. 2422–2444, 2005.

[17] J. Blanchet and F. Zhang, "Exact simulation for multivariate itô diffusions," *Advances in Applied Probability*, vol. 52, no. 4, pp. 1003–1034, 2020.

[18] D. Applebaum, *Lévy processes and stochastic calculus*. Cambridge university press, 2009.

[19] Y. Kindap and S. Godsill, "Point process simulation of generalised hyperbolic Lévy processes," *arXiv preprint arXiv:2303.10292*, 2023.

[20] S. Godsill and Y. Kındap, "Point process simulation of generalised inverse gaussian processes and estimation of the jaeger integral," *Statistics and Computing*, vol. 32, no. 1, p. 13, 2022.

[21] S. Godsill, M. Riabiz, and I. Kontoyiannis, "The Lévy state space model," in *2019 53rd Asilomar Conference on Signals, Systems, and Computers*. IEEE, 2019, pp. 487–494.

[22] E. B. Dynkin and E. B. Dynkin, *Markov processes*. Springer, 1965.

[23] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe, "Variational inference: A review for statisticians," *Journal of the American statistical Association*, vol. 112, no. 518, pp. 859–877, 2017.

[24] M. I. Jordan, "An introduction to probabilistic graphical models," 2003.

[25] D. Koller and N. Friedman, *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

[26] K. Murphy, Y. Weiss, and M. I. Jordan, "Loopy belief propagation for approximate inference: An empirical study," *arXiv preprint arXiv:1301.6725*, 2013.

[27] C. P. Robert, G. Casella, and G. Casella, *Monte Carlo statistical methods*. Springer, 1999, vol. 2.

[28] C. Zhang, J. Bütepage, H. Kjellström, and S. Mandt, "Advances in variational inference," *IEEE transactions on pattern analysis and machine intelligence*, vol. 41, no. 8, pp. 2008–2026, 2018.

[29] M. D. Hoffman, D. M. Blei, C. Wang, and J. Paisley, "Stochastic variational inference." *Journal of Machine Learning Research*, vol. 14, no. 5, 2013.

[30] R. Ranganath, S. Gerrish, and D. Blei, "Black box variational inference," in *Artificial intelligence and statistics*. PMLR, 2014, pp. 814–822.

[31] D. P. Kingma and M. Welling, "Auto-encoding variational Bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[32] K. P. Murphy, *Machine learning: a probabilistic perspective*. MIT press, 2012.

[33] S. Brooks, A. Gelman, G. Jones, and X.-L. Meng, *Handbook of Markov chain Monte carlo*. CRC press, 2011.

[34] P. Del Moral, A. Doucet, and A. Jasra, "Sequential monte carlo samplers," *Journal of the Royal Statistical Society Series B: Statistical Methodology*, vol. 68, no. 3, pp. 411–436, 2006.

[35] R. M. Neal, "Annealed importance sampling," *Statistics and computing*, vol. 11, pp. 125–139, 2001.

[36] S. Särkkä, *Bayesian filtering and smoothing*. Cambridge University Press, 2013.

[37] L. E. Baum, T. Petrie, G. Soules, and N. Weiss, "A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains," *The annals of mathematical statistics*, vol. 41, no. 1, pp. 164–171, 1970.

[38] R. E. Kalman, "A new approach to linear filtering and prediction problems," *J. Basic Eng., Trans. ASME, D*, vol. 82, pp. 35–45, 1960.

[39] B. Anderson and J. B. Moore, "Optimal filtering," *Prentice-Hall Information and System Sciences Series*, 1979.

[40] S. J. Julier and J. K. Uhlmann, "Unscented filtering and nonlinear estimation," *Proceedings of the IEEE*, vol. 92, no. 3, pp. 401–422, 2004.

[41] N. Kantas, A. Doucet, S. S. Singh, J. Maciejowski, and N. Chopin, "On particle methods for parameter estimation in state-space models," *Statistical Science*, pp. 328–351, 2015.

[42] P. Moral, *Feynman-Kac formulae: genealogical and interacting particle systems with applications*. Springer, 2004.

[43] A. Doucet, A. M. Johansen *et al.*, "A tutorial on particle filtering and smoothing: Fifteen years later," *Handbook of nonlinear filtering*, vol. 12, no. 656-704, p. 3, 2009.

[44] A. Finke, A. Doucet, and A. M. Johansen, "Limit theorems for sequential MCMC methods," *Advances in Applied Probability*, vol. 52, no. 2, pp. 377–403, 2020.

[45] A. Doucet, S. Godsill, and C. Andrieu, "On sequential Monte Carlo sampling methods for Bayesian filtering," *Statistics and computing*, vol. 10, no. 3, pp. 197–208, 2000.

[46] O. Cappé, S. J. Godsill, and E. Moulines, "An overview of existing methods and recent advances in sequential Monte Carlo," *Proceedings of the IEEE*, vol. 95, no. 5, pp. 899–924, 2007.

[47] T. Schon, F. Gustafsson, and P.-J. Nordlund, "Marginalized particle filters for mixed linear/nonlinear state-space models," *IEEE Transactions on signal processing*, vol. 53, no. 7, pp. 2279–2289, 2005.

[48] S. Godsill, "Particle filtering: the first 25 years and beyond," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 7760–7764.

[49] C. Andrieu and A. Doucet, "Particle filtering for partially observed Gaussian state space models," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 64, no. 4, pp. 827–836, 2002.

[50] Q. Li, R. Gan, J. Liang, and S. J. Godsill, "An adaptive and scalable multi-object tracker based on the non-homogeneous Poisson process," *IEEE Transactions on Signal Processing*, 2023.

[51] S. Särkkä and J. Hartikainen, "Non-linear noise adaptive Kalman filtering via variational Bayes," in *2013 IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*. IEEE, 2013, pp. 1–6.

[52] V. Šmídl and A. Quinn, *The variational Bayes method in signal processing*. Springer Science & Business Media, 2006.

[53] J. Vermaak, N. D. Lawrence, and P. Perez, "Variational inference for visual tracking," in *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings.*, vol. 1. IEEE, 2003, pp. I–I.

[54] J. Marino, M. Cvitkovic, and Y. Yue, "A general method for amortizing variational filtering," *Advances in neural information processing systems*, vol. 31, 2018.

[55] A. Campbell, Y. Shi, T. Rainforth, and A. Doucet, "Online variational filtering and parameter learning," *Advances in Neural Information Processing Systems*, vol. 34, pp. 18 633–18 645, 2021.

[56] M. Briers, A. Doucet, and S. Maskell, "Smoothing algorithms for state–space models," *Annals of the Institute of Statistical Mathematics*, vol. 62, pp. 61–89, 2010.

[57] P. Del Moral, A. Doucet, and S. Singh, "Forward smoothing using sequential monte carlo," *arXiv preprint arXiv:1012.5390*, 2010.

[58] S. J. Godsill, A. Doucet, and M. West, "Monte carlo smoothing for nonlinear time series," *Journal of the american statistical association*, vol. 99, no. 465, pp. 156–168, 2004.

[59] C. Li and S. J. Godsill, "Variational parameter learning in sequential state-space model via particle filtering," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 5589–5593.

[60] C. Andrieu, A. Doucet, and R. Holenstein, "Particle markov chain monte carlo methods," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 72, no. 3, pp. 269–342, 2010.

[61] R. Gan, J. Liang, B. I. Ahmad, and S. Godsill, "Modeling intent and destination prediction within a Bayesian framework: Predictive touch as a usecase," *Data-Centric Engineering*, vol. 1, 2020.

[62] A. J. Haug, *Bayesian Estimation and Tracking: A Practical Guide*. John Wiley & Sons, 2012.

[63] K. M. Ba h, M. G. Jæger, M. B. Skov, and N. G. Thomassen, "You can touch, but you can't look: interacting with in-vehicle systems," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 2008, pp. 1139–1148.

[64] "API overview — Leap Motion," https://developer-archive.leapmotion.com/documentation/csharp/devguide/Leap_Overview.html, accessed: 2023-09-18.

[65] F. Roider and T. Gross, "I see your point: Integrating gaze to enhance pointing gesture accuracy while driving," in *Proceedings of the 10th International Conference on Automotive User Interfaces and Interactive Vehicular Applications*. ACM, 2018, pp. 351–358.

[66] K. R. May, T. M. Gable, and B. N. Walker, "Designing an in-vehicle air gesture set using elicitation methods," in *Proceedings of the 9th International Conference on Automotive User Interfaces and Interactive Vehicular Applications*. ACM, 2017, pp. 74–83.

[67] B. I. Ahmad, J. K. Murphy, P. M. Langdon, S. J. Godsill, R. Hardy, and L. Skrypchuk, "Intent inference for hand pointing gesture-based interactions in vehicles," *IEEE transactions on cybernetics*, vol. 46, no. 4, pp. 878–889, 2015.

[68] B. I. Ahmad, J. K. Murphy, P. M. Langdon, and S. J. Godsill, "Bayesian intent prediction in object tracking using bridging distributions," *IEEE Trans. on Cybernetics*, vol. 48, no. 1, pp. 215–227, 2018.

[69] R. Gan, J. Liang, B. I. Ahmad, and S. J. Godsill, "Bayesian intent prediction for fast maneuvering objects using variable rate particle filters," in *2019 IEEE 29th International Workshop on Machine Learning for Signal Processing (MLSP)*. IEEE, 2019, pp. 1–6.

[70] J. Liang, B. I. Ahmad, R. Gan, P. Langdon, R. Hardy, and S. Godsill, "On destination prediction based on Markov bridging distributions," *IEEE Signal Processing Letters*, vol. 26, no. 11, pp. 1663–1667, Nov 2019.

[71] D. A. Castanon, B. C. Levy, and A. S. Willsky, "Algorithms for the incorporation of predictive information in surveillance theory," *International Journal of Systems Science*, vol. 16, no. 3, pp. 367–382, 1985.

[72] E. Baccarelli and R. Cusani, "Recursive filtering and smoothing for reciprocal Gaussian processes with Dirichlet boundary conditions," *IEEE Trans. on Signal Processing*, vol. 46, no. 3, pp. 790–795, 1998.

[73] L. M. Millefiori, P. Braca, K. Bryan, and P. Willett, "Modeling vessel kinematics using a stochastic mean-reverting process for long-term prediction," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 52, no. 5, pp. 2313–2330, October 2016.

[74] H. L. Christensen, J. Murphy, and S. J. Godsill, "Forecasting high-frequency futures returns using online langevin dynamics," *IEEE Journal of Selected Topics in Signal Processing*, vol. 6, no. 4, pp. 366–380, Aug 2012.

[75] M. Fanaswala and V. Krishnamurthy, "Detection of anomalous trajectory patterns in target tracking via stochastic context-free grammars and reciprocal process models," *IEEE Journal of Selected Topics in Signal Processing*, vol. 7, no. 1, pp. 76–90, 2012.

[76] R. Rezaie and X. R. Li, "Gaussian conditionally Markov sequences: Dynamic models and representations of reciprocal and other classes," *IEEE Transactions on Signal Processing*, vol. 68, pp. 155–169, 2019.

[77] T. Bando, K. Takenaka, S. Nagasaka, and T. Taniguchi, "Unsupervised drive topic finding from driving behavioral data," in *Proc. of IEEE Intelligent Vehicles Symposium (IV)*, 2013, pp. 177–182.

[78] B. Völz, H. Mielenz, I. Gilitschenski, R. Siegwart, and J. Nieto, "Inferring pedestrian motions at urban crosswalks," *IEEE Transactions on Intelligent Transportation Systems*, 2018.

[79] S. Gaurav and B. Ziebart, "Discriminatively learning inverse optimal control models for predicting human intentions," in *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*, ser. AAMAS '19. Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems, 2019, pp. 1368–1376. [Online]. Available: http://dl.acm.org/citation.cfm?id=3306127.3331844

[80] C. M. Harris and D. M. Wolpert, "Signal-dependent noise determines motor planning," *Nature*, vol. 394, no. 6695, p. 780, 1998.

[81] S. Godsill, "Particle filters for continuous-time jump models in tracking applications," *ESAIM: Proc.*, vol. 19, pp. 39–52, 2007. [Online]. Available: https://doi.org/10.1051/proc:071907

[82] R. N. Clark, "The Routh-Hurwitz stability criterion, revisited," *IEEE Control Systems Magazine*, vol. 12, no. 3, pp. 119–120, 1992.

[83] D. Gasbarra, T. Sottinen, and E. Valkeila, "Gaussian bridges," in *Stochastic analysis and applications*. Springer, 2007, pp. 361–382.

[84] G. Zhou, K. Li, and T. Kirubarajan, "Constrained state estimation using noisy destination information," *Signal Processing*, vol. 166, p. 107226, 2020. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0165168419302725

[85] S. G. Kou, "A jump-diffusion model for option pricing," *Management science*, vol. 48, no. 8, pp. 1086–1101, 2002.

[86] B. I. Ahmad, J. Murphy, P. M. Langdon, and S. J. Godsill, "Filtering perturbed in-vehicle pointing gesture trajectories: Improving the reliability of intent inference," in *2014 IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, Sep. 2014, pp. 1–6.

[87] A. C. Harvey, *Forecasting, structural time series models and the Kalman filter*. Cambridge university press, 1990.

[88] S. J. Godsill, J. Vermaak, W. Ng, and J. F. Li, "Models and algorithms for tracking of maneuvering objects using variable rate particle filters," *Proceedings of the IEEE*, vol. 95, no. 5, pp. 925–952, 2007.

[89] "Ergonomics of human-system interaction — Part 411: Evaluation methods for the design of physical input devices," International Organization for Standardization, Geneva, CH, Standard, May 2012.

[90] B. I. Ahmad, P. M. Langdon, P. Bunch, and S. J. Godsill, "Probabilistic intentionality prediction for target selection based on partial cursor tracks," in *International Conference on Universal Access in Human-Computer Interaction*. Springer, 2014, pp. 427–438.

[91] R. Gan and S. J. Godsill, "α-stable Lévy state-space models for manoeuvring object tracking," *2020 IEEE 23rd International Conference on Information Fusion (FUSION)*, pp. 1–7, 2020.

[92] J. Liang, B. I. Ahmad, and S. Godsill, "Simultaneous intent prediction and state estimation using an intent-driven intrinsic coordinate model," in *2020 IEEE 30th International Workshop on Machine Learning for Signal Processing (MLSP)*, 2020.

[93] R. A. Singer, "Estimating optimal tracking filter performance for manned maneuvering targets," *IEEE Transactions on Aerospace and Electronic Systems*, vol. AES-6, no. 4, pp. 473–483, July 1970.

[94] R. Gan, B. I. Ahmad, and S. J. Godsill, "Lévy state-space models for tracking and intent prediction of highly maneuverable objects," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 57, no. 4, 2021.

[95] R. Gan and S. Godsill, "α-stable Lévy state-space models for manoeuvring object tracking," in *2020 IEEE 23rd International Conference on Information Fusion (FUSION)*. IEEE, 2020, pp. 1–7.

[96] S. Särkkä, *Recursive Bayesian inference on stochastic differential equations*. Helsinki University of Technology, 2006.

[97] M. H. Dickinson and F. T. Muijres, "The aerodynamics and control of free flight manoeuvres in drosophila," *Philosophical Transactions of the Royal Society B: Biological Sciences*, vol. 371, no. 1704, p. 20150388, 2016.

[98] A. M. Reynolds and M. A. Frye, "Free-flight odor tracking in drosophila is consistent with an optimal intermittent scale-free search," *PloS one*, vol. 2, no. 4, p. e354, 2007.

[99] M. Shao and C. L. Nikias, "Signal processing with fractional lower order moments: stable processes and their applications," *Proceedings of the IEEE*, vol. 81, no. 7, pp. 986–1010, 1993.

[100] G. Samoradnitsky and M. Taqqu, *Stable non-Gaussian random processes: Stochastic models with infinite variance*, 1st ed. Chapman and Hall, 06 1994.

[101] B. V. Gnedenko and A. N. Kolmogorov, *Limit distributions for sums of independent random variables*. Addison-Wesley, 1968.

[102] M. L. de Freitas, M. Egan, L. Clavier, A. Goupil, G. W. Peters, and N. Azzaoui, "Capacity bounds for additive symmetric α-stable noise channels," *IEEE Trans. on Information Theory*, vol. 63, no. 8, pp. 5115–5123, Aug 2017.

[103] P. G. Georgiou, P. Tsakalides, and C. Kyriakakis, "Alpha-stable modeling of noise and robust time-delay estimation in the presence of impulsive noise," *IEEE Trans. on Multimedia*, vol. 1, no. 3, pp. 291–301, 1999.

[104] N. Azzaoui and L. Clavier, "Statistical channel model based on α-stable random processes and application to the 60 ghz ultra wide band channel," *IEEE Trans. on Communications*, vol. 58, no. 5, pp. 1457–1467, 2010.

[105] S. Godsill and P. Rayner, *Digital Audio Restoration - a statistical model based approach*. Springer, September 1998.

[106] W. Schoutens, *Lévy processes in finance: pricing financial derivatives*. John Wiley & Sons, 2003.

[107] T. Lemke, M. Riabiz, and S. J. Godsill, "Fully Bayesian inference for α-stable distributions using a Poisson series representation," *Digital Signal Processing*, vol. 47, pp. 96–115, 2015.

[108] S. S. Lim and M. Farooq, "Maneuvering target tracking using jump processes," in *Proceedings of the 30th IEEE Conference on Decision and Control*, Dec 1991, pp. 2049–2054 vol.2.

[109] N. Gordon, J. Percival, and M. Robinson, "The Kalman-Lévy filter and heavy-tailed models for tracking manoeuvring targets," in *Proc. 6th Int. Conf. on Information Fusion*, 2003, pp. 1024–1031.

[110] S. Godsill and E. Kuruoglu, "Bayesian inference for time series with heavy-tailed symmetric α-stable noise processes," in *In Proc. Applications of heavy tailed distributions in economics, engineering and statistics*, 1999.

[111] T. Lemke and S. J. Godsill, "Enhanced Poisson sum representation for alpha-stable processes," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, May 2011, pp. 4100–4103.

[112] M. Riabiz and S. J. Godsill, "Approximate simulation of linear continuous time models driven by asymmetric stable Lévy processes," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017, pp. 4676–4680.

[113] T. Lemke and S. Godsill, *Inference for models with asymmetric α-stable noise processes*. Oxford University Press, 2015, ch. 9.

[114] M. Riabiz, T. Ardeshiri, I. Kontoyiannis, and S. Godsill, "Nonasymptotic Gaussian approximation for inference with stable noise," *IEEE Transactions on Information Theory*, 2020.

[115] O. C. Ibe, *Elements of random walk and diffusion processes*. John Wiley & Sons, 2013.

[116] A. M. Reynolds and C. J. Rhodes, "The Lévy flight paradigm: random search patterns and mechanisms," *Ecology*, vol. 90, no. 4, pp. 877–887, 2009.

[117] F. Bartumeus, "Lévy processes in animal movement: an evolutionary hypothesis," *Fractals*, vol. 15, no. 02, pp. 151–162, 2007.

[118] N. E. Humphries, H. Weimerskirch, N. Queiroz, E. J. Southall, and D. W. Sims, "Foraging success of biological Lévy flights recorded in situ," *Proceedings of the National Academy of Sciences*, vol. 109, no. 19, pp. 7169–7174, 2012.

[119] V. Krishnamurthy and S. Gao, "Syntactic enhancement to vsimm for roadmap based anomalous trajectory detection: A natural language processing approach," *IEEE Transactions on Signal Processing*, vol. 66, no. 20, pp. 5212–5227, 2018.

[120] M. Riabiz, "On latent variable models for Bayesian inference with stable distributions and processes," Ph.D. dissertation, University of Cambridge, 2019.

[121] J. Rosiński, "Series representations of Lévy processes from the perspective of point processes," in *Lévy Processes: Theory and Applications*. Springer, 2001, pp. 401–415.

[122] M. S. Grewal and A. P. Andrews, *Kalman filtering: Theory and Practice with MATLAB*. John Wiley & Sons, 2014.

[123] M. J. McGuffin and R. Balakrishnan, "Fitts' law and expanding targets: Experimental studies and designs for user interfaces," *ACM Transactions on Computer-Human Interaction*, vol. 12, no. 4, pp. 388–422, 2005.

[124] T. Hastie, R. Tibshirani, J. H. Friedman, and J. H. Friedman, *The elements of statistical learning: data mining, inference, and prediction*. Springer, 2009, vol. 2.

[125] J. H. Ramos Zuniga, "Extending the practical applicability of the kalman filter," Ph.D. dissertation, 2020.

[126] H. E. Rauch, F. Tung, and C. T. Striebel, "Maximum likelihood estimates of linear dynamic systems," *AIAA journal*, vol. 3, no. 8, pp. 1445–1450, 1965.

[127] R. Gan and S. Godsill, "Conditionally factorized variational Bayes with importance sampling," in *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2022, pp. 5777–5781.

[128] M. J. Beal, "Variational algorithms for approximate Bayesian inference," Ph.D. dissertation, Gatsby Computational Neuroscience Unit, University College London, 2003. [Online]. Available: http://www.cse.buffalo.edu/faculty/mbeal/thesis/index.html

[129] V. Smidl and A. Quinn, "Variational Bayesian filtering," *IEEE Transactions on Signal Processing*, vol. 56, no. 10, pp. 5020–5030, 2008.

[130] S. Sarkka and A. Nummenmaa, "Recursive noise adaptive Kalman filtering by variational Bayesian approximations," *IEEE Transactions on Automatic control*, vol. 54, no. 3, pp. 596–600, 2009.

[131] T. Ardeshiri, E. Özkan, U. Orguner, and F. Gustafsson, "Approximate Bayesian smoothing with unknown process and measurement noise covariances," *IEEE Signal Processing Letters*, vol. 22, no. 12, pp. 2450–2454, 2015.

[132] M. D. Hoffman and D. M. Blei, "Structured stochastic variational inference," in *Artificial Intelligence and Statistics*, 2015, pp. 361–369.

[133] L. Ye, A. Beskos, M. De Iorio, and J. Hao, "Monte Carlo co-ordinate ascent variational inference," *Statistics and Computing*, pp. 1–19, 2020.

[134] L. K. Saul and M. I. Jordan, "Exploiting tractable substructures in intractable networks," *Advances in neural information processing systems*, pp. 486–492, 1996.

[135] P. Carbonetto and N. De Freitas, "Conditional mean field," in *Advances in Neural Information Processing Systems 19: Proceedings of the 2006 Conference*, vol. 19. MIT Press, 2007, p. 201.

[136] T. S. Jaakkola and M. I. Jordan, "Improving the mean field approximation via the use of mixture distributions," in *Learning in graphical models*. Springer, 1998, pp. 163–173.

[137] J. Y. Liu and X. Qiao, "Conditional variational inference with adaptive truncation for Bayesian nonparametric models," *arXiv preprint arXiv:2001.04508*, 2020.

[138] P. J. Cameron, *Naïve set theory*. London: Springer London, 1998, pp. 1–36. [Online]. Available: https://doi.org/10.1007/978-1-4471-0589-3_1

[139] Y. Burda, R. Grosse, and R. Salakhutdinov, "Importance weighted autoencoders," *arXiv preprint arXiv:1509.00519*, 2015.

[140] C. Naesseth, S. Linderman, R. Ranganath, and D. Blei, "Variational sequential Monte Carlo," in *International conference on artificial intelligence and statistics*. PMLR, 2018, pp. 968–977.

[141] C. J. Maddison, D. Lawson, G. Tucker, N. Heess, M. Norouzi, A. Mnih, A. Doucet, and Y. W. Teh, "Filtering variational objectives," *arXiv preprint arXiv:1705.09279*, 2017.

[142] Q. Li, B. I. Ahmad, and S. J. Godsill, "Sequential dynamic leadership inference using Bayesian Monte Carlo methods," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 57, no. 4, pp. 2039–2052, 2021.

[143] R. Gan, Q. Li, and S. Godsill, "A variational Bayes association-based multi-object tracker under the non-homogeneous Poisson measurement process," in *2022 25th International Conference on Information Fusion (FUSION)*. IEEE, 2022, pp. 1–8.

[144] K. Gilholm, S. Godsill, S. Maskell, and D. Salmond, "Poisson models for extended target and group tracking," in *Signal and Data Processing of Small Targets 2005*, vol. 5913. International Society for Optics and Photonics, 2005, p. 59130R.

[145] F. Septier, S. K. Pang, A. Carmi, and S. Godsill, "On MCMC-based particle methods for Bayesian filtering: Application to multitarget tracking," in *2009 3rd IEEE International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP)*. IEEE, 2009, pp. 360–363.

[146] Q. Li, J. Liang, and S. Godsill, "Scalable data association and multi-target tracking under a Poisson mixture measurement process," in *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2022, pp. 5503–5507.

[147] S. Yang, K. Thormann, and M. Baum, "Linear-time joint probabilistic data association for multiple extended object tracking," in *2018 IEEE 10th Sensor Array and Multichannel Signal Processing Workshop (SAM)*. IEEE, 2018, pp. 6–10.

[148] Q. Li, J. Sun, and W. Sun, "An efficient multiple hypothesis tracker using max product belief propagation," in *2017 20th International Conference on Information Fusion (Fusion)*. IEEE, 2017, pp. 1–6.

[149] F. Meyer, T. Kropfreiter, J. L. Williams, R. Lau, F. Hlawatsch, P. Braca, and M. Z. Win, "Message passing algorithms for scalable multitarget tracking," *Proceedings of the IEEE*, vol. 106, no. 2, pp. 221–259, 2018.

[150] F. Meyer and M. Z. Win, "Scalable data association for extended object tracking," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 6, pp. 491–507, 2020.

[151] R. L. Streit and T. E. Luginbuhl, "Probabilistic multi-hypothesis tracking," Naval Underwater Systems Center Newport RI, Tech. Rep., 1995.

[152] Q. Li, S. J. Godsill, J. Liang, and B. I. Ahmad, "Inferring dynamic group leadership using sequential Bayesian methods," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 8911–8915.

[153] R. Gan, Q. Li, and S. Godsill, "Variational tracking and redetection for closely-spaced objects in heavy clutter," *arXiv preprint arXiv:2309.01774*, 2023.

[154] K. Granström, M. Fatemi, and L. Svensson, "Poisson multi-Bernoulli mixture conjugate prior for multiple extended target filtering," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 56, no. 1, pp. 208–225, 2019.

[155] K. Granström, L. Svensson, S. Reuter, Y. Xia, and M. Fatemi, "Likelihood-based data association for extended object tracking using sampling methods," *IEEE Transactions on intelligent vehicles*, vol. 3, no. 1, pp. 30–45, 2017.

[156] F. Meyer and J. L. Williams, "Scalable detection and tracking of geometric extended objects," *IEEE Transactions on Signal Processing*, vol. 69, pp. 6283–6298, 2021.

[157] J. Williams and R. Lau, "Approximate evaluation of marginal association probabilities with belief propagation," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 50, no. 4, pp. 2942–2959, 2014.

[158] C.-Y. Chong, "Graph approaches for data association," in *2012 15th international conference on information fusion*. IEEE, 2012, pp. 1578–1585.

[159] D. Cormack and J. R. Hopgood, "Message passing and hierarchical models for simultaneous tracking and registration," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 57, no. 3, pp. 1524–1537, 2021.

[160] R. Mahler, "Multitarget Bayes filtering via first-order multitarget moments," *IEEE Transactions on Aerospace and Electronic systems*, vol. 39, no. 4, pp. 1152–1178, 2003.

[161] R. D. Turner, S. Bottone, and B. Avasarala, "A complete variational tracker," *Advances in Neural Information Processing Systems*, vol. 27, 2014.

[162] R. A. Lau and J. L. Williams, "A structured mean field approach for existence-based multiple target tracking," in *2016 19th International Conference on Information Fusion (FUSION)*. IEEE, 2016, pp. 1111–1118.

[163] D. Schuhmacher, B.-T. Vo, and B.-N. Vo, "A consistent metric for performance evaluation of multi-object filters," *IEEE transactions on signal processing*, vol. 56, no. 8, pp. 3447–3457, 2008.

[164] Q. Li and S. Godsill, "A new leader-follower model for bayesian tracking," in *2020 IEEE 23rd International Conference on Information Fusion (FUSION)*. IEEE, 2020, pp. 1–7.

[165] L. Mihaylova, A. Y. Carmi, F. Septier, A. Gning, S. K. Pang, and S. Godsill, "Overview of bayesian sequential monte carlo methods for group and extended object tracking," *Digital Signal Processing*, vol. 25, pp. 1–16, 2014.

[166] Q. Li, R. Gan, and S. Godsill, "A scalable rao-blackwellised sequential mcmc sampler for joint detection and tracking in clutter," in *2023 26th International Conference on Information Fusion (FUSION)*. IEEE, 2023, pp. 1–8.

[167] F. Goodyer, B. I. Ahmad, and S. Godsill, "Flexible multi-target tracking with track management using dirichlet and gaussian processes," in *2023 26th International Conference on Information Fusion (FUSION)*. IEEE, 2023, pp. 1–8.

[168] G. Battistelli, L. Chisci, G. Mugnai, A. Farina, and A. Graziano, "Consensus-based linear and nonlinear filtering," *IEEE Transactions on Automatic Control*, vol. 60, no. 5, pp. 1410–1415, 2014.

[169] Q. Li, R. Gan, and S. Godsill, "Consensus-based distributed variational multi-object tracker in multi-sensor network," in *2023 Sensor Signal Processing for Defence Conference (SSPD)*. IEEE, 2023, pp. 1–5.

[170] J. Liang, B. I. Ahmad, M. Jahangir, and S. Godsill, "Detection of malicious intent in non-cooperative drone surveillance," in *2021 Sensor Signal Processing for Defence Conference (SSPD)*. IEEE, 2021, pp. 1–5.