Linköping Studies in Science and Technology. Dissertations
No. 579

Recursive Bayesian Estimation
Navigation and Tracking Applications

Niclas Bergman

Department of Electrical Engineering
Linköping University, SE–581 83 Linköping, Sweden

Linköping 1999

Cover Illustration: The front cover depicts a terrain elevation map covering a southern part of the province of Östergötland, Sweden. Bright color indicates a high terrain elevation value. The corresponding terrain information, or variation, for the same area is shown on the back cover. The visible large dark areas are the lakes Sommen and Åsunden. The Baltic sea and the archipelago of Västervik are visible in the east part of the map. The terrain information on the cover is part of a large map used in the terrain navigation application, presented in Chapter 2 and Chapter 7. Similar illustrations with further explanations are given on page 8 of the thesis.

Recursive Bayesian Estimation:
Navigation and Tracking Applications

niclas@isy.liu.se
http://www.control.isy.liu.se
Department of Electrical Engineering
Linköping University
SE–581 83 Linköping
Sweden

ISBN 91-7219-473-1 ISSN 0345-7524

Printed in Sweden by Linus & Linea AB

Abstract

Recursive estimation deals with the problem of extracting information about parameters, or states, of a dynamical system in real time, given noisy measurements of the system output. Recursive estimation plays a central role in many applications of signal processing, system identification and automatic control. In this thesis we study nonlinear and non-Gaussian recursive estimation problems in discrete time. Our interest in these problems stems from the airborne applications of target tracking, and autonomous aircraft navigation using terrain information.

In the Bayesian framework of recursive estimation, both the sought parameters and the observations are considered as stochastic processes. The conceptual solution to the estimation problem is found as a recursive expression for the posterior probability density function of the parameters conditioned on the observed measurements. This optimal solution to nonlinear recursive estimation is usually impossible to compute in practice, since it involves several integrals that lack analytical solutions.

We phrase the application of terrain navigation in the Bayesian framework, and develop a numerical approximation to the optimal but intractable recursive solution. The designed point-mass filter computes a discretized version of the posterior filter density in a uniform mesh over the interesting region of the parameter space. Both the uniform mesh resolution and the grid point locations are automatically adjusted at each iteration of the algorithm. This Bayesian point-mass solution is shown to yield high navigation performance in a simulated realistic environment.

Even though the optimal Bayesian solution is intractable to implement, the performance of the optimal solution is assessable and can be used for comparative evaluation of suboptimal implementations. We derive explicit expressions for the Cramér-Rao bound of general nonlinear filtering, smoothing and prediction problems. We consider both the cases of random and nonrandom modeling of the parameters. The bounds are recursively expressed and are connected to linear recursive estimation. The newly developed Cramér-Rao bounds are applied to the terrain navigation problem, and the point-mass filter is verified to reach the bound in exhaustive simulations.

The uniform mesh of the point-mass filter limits it to estimation problems of low dimension. Monte Carlo methods offer an alternative approach to recursive estimation and promise tractable solutions to general high dimensional estimation problems. We provide a review over the active field of statistical Monte Carlo methods. In particular, we study the particle filters for recursive estimation. Three different particle filters are applied to terrain navigation, and evaluated against the Cramér-Rao bound and the point-mass filter. The particle filters utilize an adaptive grid representation of the filter density and are shown to yield a performance equal to the point-mass method.

A Markov Chain Monte Carlo (MCMC) method is developed for a highly complex data association problem in target tracking. This algorithm is compared to previously proposed methods and is shown to yield competitive results in a simulation study.

Preface

Some of the results of this thesis have, or will, appear as published material. The work was initially focused on the application of terrain navigation, which resulted in the conference papers

[15] N. Bergman. A Bayesian approach to terrain-aided navigation. In *Proc. of SYSID'97, 11th IFAC Symposium on System Identification*, pages 1531–1536. IFAC, 1997,

[23] N. Bergman, L. Ljung, and F. Gustafsson. Point-mass filter and Cramér-Rao bound for terrain-aided navigation. In *Proc. 36:th IEEE Conf. on decision and control*, pages 565–570, 1997

and the Licentiate thesis

[16] N. Bergman. *Bayesian Inference in Terrain Navigation.* Linköping Studies in Science and Technology. Thesis No 649, 1997.

Most results on terrain navigation presented in this thesis can be found in [16]. The terrain navigation application nicely encompasses the major concepts of the general problem of nonlinear recursive estimation, and is therefore used as an extensive introduction to the thesis presented in Chapter 2. This chapter is an exact copy of the forthcoming article

[24] N. Bergman, L. Ljung, and F. Gustafsson. Terrain navigation using Bayesian statistics. *IEEE Control Systems Magazine*, 19(3), June 1999.

The results on Cramér-Rao bounds presented in Chapter 4 and Section 7.2 are covered by

[23] N. Bergman, L. Ljung, and F. Gustafsson. Point-mass filter and Cramér-Rao bound for terrain-aided navigation. In *Proc. 36:th IEEE Conf. on decision and control*, pages 565–570, 1997.

[25] N. Bergman and P. Tichavský. Two Cramér-Rao bounds for terrain-aided navigation. *IEEE Transactions on Aerospace and Electronic Systems*, 1999. In review.

The wavelet approach to spatial grid adaptation of Section 5.3 has been presented as

[20] N. Bergman. An interpolating wavelet filter for terrain navigation. In *Proc. conf. on multisource-multisensor information fusion*, pages 251–258, 1998.

The Monte Carlo methods for terrain navigation given in Section 7.4 have, and will, appear as

[18] N. Bergman. Deterministic and stochastic Bayesian methods in terrain navigation. In *Proc. 37:th IEEE Conf. on Decision and Control*, 1998.

[21] N. Bergman. Terrain navigation using sequential Monte Carlo methods. In A. Doucet, J.F.G. de Freitas, and N.J. Gordon, editors, *Sequential Monte Carlo methods in practice*. Cambridge University Press, 1999. To appear.

Part of the results on target tracking in Chapter 8 will be published in

[22] N. Bergman and F. Gustafsson. Three statistical batch algorithms for tracking manoeuvring targets. In *Proc. 5th European Control Conference*, Karlsruhe, Germany, 1999.

Material not directly covered by the thesis, but nevertheless related to the concepts in Chapter 8, is found in

[93] A. Isaksson, F. Gustafsson, and N. Bergman. Pruning versus merging in Kalman filter banks for manoeuvre tracking. *IEEE Transactions on Aerospace and Electronic Systems*, 1999. Accepted for publication.

Acknowledgments

My supervisors Professor Fredrik Gustafsson and Professor Lennart Ljung deserve my deepest gratitude. This thesis is to a large extent a results of their skillful guidance and encouraging support of my work. I am extra grateful to Fredrik for all the fruitful, spontaneous discussions we have had during this work.

Several other distinguished researchers have supported me in this work. I would like to thank Arnaud Doucet for introducing me to the field of statistical Monte Carlo methods, and taking care of me during my visit to Cambridge. Petr Tichavský is acknowledged for introducing me to the posterior Cramér-Rao bound, and Andrew Logothetis for sharing the tracking application with me.

Parts of the manuscript have at different stages been reviewed by Arnaud Doucet, Fredrik Tjärnström, Fredrik Gunnarsson, and Urban Forssell. Their valuable comments and suggestions have lifted the thesis to a level I never would have reached on my own.

This work has been partly sponsored by ISIS, a NUTEK Competence Center at Linköping University. The simulation data and the terrain database have been provided by Saab Dynamics, Linköping Sweden. Detailed explanations about the navigation application have also been provided by several employees of Saab, Linköping.

Linköping, April 1999

Niclas Bergman

Contents

1

Introduction

Nonlinear and non-Gaussian recursive estimation problems are commonly encountered, e.g., in target tracking and navigation applications. Practical solutions to these problems usually resort to model approximations, applying standard linear solutions to local linearizations of the estimation model. This will yield algorithms with modest computational requirements, but will clearly be suboptimal when the nonlinearities of the problem are severe. The local linearization schemes therefore fail in applications where detailed nonlinear models are available and the estimation performance cannot be sacrificed.

In this work, we emphasize the Bayesian view on statistical inference and review the tools available for approximative implementation of this approach to nonlinear recursive estimation. Instead of describing the problem model approximatively, the optimal solution is here implemented in an approximative fashion.

1.1 Thesis Outline

The next chapter is an exact copy of the submitted final manuscript to [24]. This article highlights the major problems of recursive estimation by studying the terrain navigation application in detail. It is this application that has been the foundational application, propelling the work presented in this thesis. The chapter contains a brief presentation of the terrain navigation application, its Bayesian solution, and

an approximative numerical integration method developed with this application in mind. This introduction serves both to exemplify the type of problems we consider, and to present the main results obtained for the terrain navigation application. There will inevitably be some overlap between Chapter 2 and the rest of the thesis, in particular when we return to the terrain navigation application in Chapter 7.

Chapter 3 contains a comprehensive review over estimation theory with an emphasis on the Bayesian paradigm of statistical inference. The review aims at describing the Bayesian view of statistical inference in general, and its application to the recursive estimation problem in particular. The chapter also contains comments on optimal combination of sensor measurements and clarifies some connections between classical statistical inference and the inference method of Dempster–Shafer. A review over approaches to nonlinear recursive estimation is also given in Chapter 3.

In Chapter 4 we present bounds on the mean square estimation error in recursive estimation. Firstly, we review and extend the basic results of Cramér-Rao bounds for both random and parametric estimation. Secondly, these bounds are applied to the recursive estimation problem. We derive recursive expressions for the Cramér-Rao bounds to filtering prediction and smoothing in nonlinear, non-Gaussian state space models. The presentation is streamlined by emphasizing the obtained results and postponing all proofs to appendices to the chapter.

In Chapter 5, we consider a direct approach to implementing the optimal Bayesian solution by means of numerical integration. This leads to grid based methods that recursively computes a discretized version of the posterior filter density. We exemplify such methods by presenting a grid based method specifically developed for the terrain navigation application. We also comment the difficulties in applying grid based methods to problems of high dimension, and presents some results on adapting the grid spatially.

Chapter 6 contains a review over simulation based methods, which are better suited for high dimensional problems. In these Monte Carlo methods, statistical inference is performed by simulating a large number of candidate parameters and computing sample averages with respect to these. We review the general concept of Monte Carlo methods and present the basic theoretical justifications for their properties along with several algorithms. Most importantly, we present the Monte Carlo methods for recursive estimation.

In Chapter 7, we return to the terrain navigation application and describe the integration between terrain navigation and inertial navigation. Different approaches to terrain navigation are reviewed and compared to the Bayesian approach suggested in this work. We combine the grid based method for Bayesian terrain navigation of Chapter 5 with the Cramér-Rao bounds of Chapter 4. In extensive simulations we show that the implementation reach a nearly optimal performance level with a densely chosen grid resolution. Comparative simulation studies between the grid based method, Cramér-Rao bound, and some algorithms from Chapter 6 are also provided.

Chapter 8 contains applications of Bayesian inference to problems in target tracking. Firstly, we review the Expectation Maximization (EM) algorithm and

apply it to a target manoeuvre detection problem. Secondly, we study a complex data association problem occuring in over the horizon target tracking. We develop a Gibbs sampling procedure for this problem and compare it to two other previously suggested approaches. The applications of target tracking that we consider are off-line algorithms and therefore fall slightly out of scope, judging form the title of the thesis. Chapter 8 instead serves to illustrate Bayesian techniques in general, and some concepts of Chapter 6 in particular.

Finally, Chapter 9 provides some conclusive remarks regarding the work, and possible directions of future research.

1.2 Contributions

Chapter 3 and Chapter 6 contain overviews of statistical Bayesian estimation and numerical Monte Carlo methods, respectively. The complementary chapters contain the main contributions of the thesis, they are listed below in order of appearance.

- The Bayesian formulation of, and solution to, the terrain navigation problem, presented in Chapter 2.
- The parametric Cramér-Rao bounds for filtering, prediction, and smoothing in nonlinear, and non-Gaussian, recursive state space estimation problems. The bounds are presented in Chapter 4.
- The posterior Cramér-Rao bounds for singular state evolution and for smoothing in nonlinear and non-Gaussian recursive estimation, also given in Chapter 4.
- The numerical integration point-mass filter developed for the terrain navigation, detailed in Chapter 5.
- The Cramér-Rao bounds and particle filters for terrain navigation, presented in Chapter 7.
- The Gibbs sampling procedure for multiple measurement data association, described in Chapter 8.

1.3 Reading Directions

Readers primarily interested in the terrain navigation application may read Chapter 2, and thereafter resort to Section 5.2 and Chapter 7. Complementary information about the intricate details of the terrain navigation application can be found in the Licentiate thesis [16]. However, the Bayesian approach to this problem, and its solution, is fully covered in this work.

Readers with an aim to solely study the parts on the Cramér-Rao bounds for recursive estimation are suggested to go directly to Chapter 4. The basic concepts

of estimation theory can be found in Chapter 3 if a prior knowledge of statistical estimation is lacking. Illustrations of the practical application of the bounds are found in Section 7.2, Section 7.3, and Section 7.4.

The review of Monte Carlo methods in Chapter 6 is to a large extent self-contained and may be read without reference to the other chapters. However, it will prove beneficial to consult Section 7.4 and Section 8.2 to exemplify and illuminate the concepts presented in Chapter 6.

2

Terrain Navigation using Bayesian Statistics

In aircraft navigation the demands on reliability and safety are very high. The importance of accurate position and velocity information becomes crucial when flying an aircraft at low altitudes, and especially during the landing phase. Not only should the navigation system have a consistent description of the position of the aircraft, but also a description of the surrounding terrain, buildings and other objects that are close to the aircraft. Terrain navigation is a navigation scheme that utilizes variations in the terrain height along the aircraft flight path. Integrated with an Inertial Navigation System (INS), it yields high performance position estimates in an autonomous manner, i.e., without any support information sent to the aircraft. In order to obtain these position estimates, a nonlinear recursive estimation problem must be solved on-line. Traditionally, this filtering problem has been solved by local linearization of the terrain at one or several assumed aircraft positions. Due to changing terrain characteristics, these linearizations will in some cases result in diverging position estimates. In this work, we show how the Bayesian approach gives a comprehensive framework for solving the recursive estimation problem in terrain navigation. Instead of approximating the model of the estimation problem, the analytical solution is approximately implemented. The proposed navigation filter computes a probability mass distribution of the aircraft position and updates this description recursively with each new measurement. The navigation filter is evaluated over a commercial terrain database, yielding accurate position estimates over several types of terrain characteristics. Moreover, in a

Monte Carlo analysis, it shows optimal performance as it reaches the Cramér-Rao lower bound.

2.1 Aircraft Navigation

Navigation is the concept of determination of the kinematic state of a moving vehicle. In aircraft navigation this usually consists of finding the position and velocity of the aircraft. Accurate knowledge of this state is critical for flight safety. Therefore, an aircraft navigation system should not only provide a reliable and accurate estimate of the current kinematic state of the aircraft, but also a consistent description of the accuracy of this estimate.

Aircraft navigation is typically performed using a combination of dead-reckoning and fix position updates. In dead-reckoning systems, the state vector is calculated from a continuous series of measurements of the aircraft movement relative to an initial position. Due to error accumulation, dead-reckoning systems must be re-initialized periodically. Fix point, or positioning, systems measure the state vector more or less without regard to the previous movement of the aircraft. They are therefore suitable for re-initialization of dead-reckoning systems.

The most common dead-reckoning systems are the Inertial Navigation Systems (INS) in which accelerometers are used to sense the magnitude of the aircraft acceleration. A set of gyroscopes either maintains the accelerometers in a known orientation with respect to a fixed, non-rotating coordinate system, commonly referred to as inertial space, or measures the angular rate of the accelerometers relative to inertial space. The inertial navigation computer uses these sensed accelerations and angular rates to compute the aircraft velocity, position, attitude, attitude rate, heading, altitude, and possibly range and bearing to destination. An INS generates near instantaneous continuous position and velocity, it is self-contained, functions at all latitudes, and in all weather conditions. It operates independently of aircraft manoeuvres and without the need for ground station support. Complete and comprehensive presentations of inertial navigation can be found in [99, 135].

Positioning systems that have attracted a lot of attention lately are the global satellite navigation systems which promise a very high accuracy and global coverage. There are two global satellite systems for navigation in use today: GPS, developed by the U.S., and the Russian system GLONASS. Position estimates are obtained by comparing distances from the aircraft to four or more satellites. The systems have been developed for military purposes and several coding techniques are used to keep the accuracy for civilian or unauthorized users at a level far from the actual performance of the systems. However, using ground stations as reference, the coding errors can be removed efficiently. Vendors have off-the-shelf receivers for differential GPS (DGPS) with a position accuracy below the one meter level. A comprehensive summary of the concept of satellite navigation can be found in [99, Chapter 5].

The radio navigation systems have the disadvantage of relying on information broadcasted to the aircraft. This information could be deliberately jammed in

a hostile situation, or the transmitters could be destroyed, leaving the aircraft without navigation support. Hence, even if the satellite systems give high accuracy position information they need to be combined with alternative backup systems using other navigation principles. The concept of terrain navigation is an alternative positioning technique that autonomously generates updates to the INS, although in general not with the same accuracy as the satellite systems. The main idea in terrain navigation is to measure the variations in the terrain height underneath the aircraft flight path and compare these measurements with a reference map. The

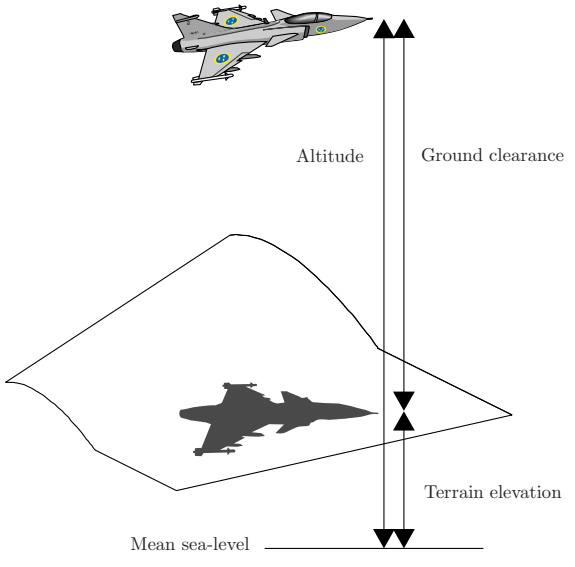


Figure 2.1: The principle of terrain navigation.

principle of terrain navigation is depicted in Figure 2.1. The aircraft altitude over mean sea-level is measured with a barometric altimeter and the ground clearance is measured with a radar altimeter, pointing downward. The terrain elevation beneath the aircraft is found by taking the difference between the altitude and ground clearance measurements. The navigation computer holds a digital reference map with values of the terrain elevation as a function of longitude and latitude. The measured terrain elevation is compared with this reference map and matching positions in the map are determined. Terrain repetitiveness and flatness make this matching nontrivial and the quality of the outcome dependent on the amount of terrain variation. Many areas inside the reference map will in general have a terrain elevation comparable to the measured one. In order to distinguish the true position from false ones, several measurements along the aircraft flight path need to be considered. Hence, the measurements must be matched with the map on-line and in a recursive manner. For comprehensive discussions about the applications of terrain navigation techniques, see [86, 87].

The performance of the matching of terrain elevation measurements with the map depends highly on the type of terrain in the area. Flat terrain gives little or no information about the aircraft position. Rough, but repetitive, terrain can give several well matched positions in an area, making it hard to distinguish between several, well matching tracks. The information content inside a generic area of the map can be shown to be proportional to the average size of the terrain gradient,

$$\sqrt{\frac{1}{N}\sum_{i=1}^{N}\|\nabla h(x_i)\|^2} \tag{2.1}$$

where x_i are positions uniformly distributed in the area of interest. This scalar measure of the terrain information can be connected to an associated Cramér-Rao bound for the underlying estimation problem [16]. The right part of Figure 2.2 shows (2.1) evaluated in square blocks of 400 meter side where bright color indicates a large value. The terrain map used in this work is a real commercial map of a 100

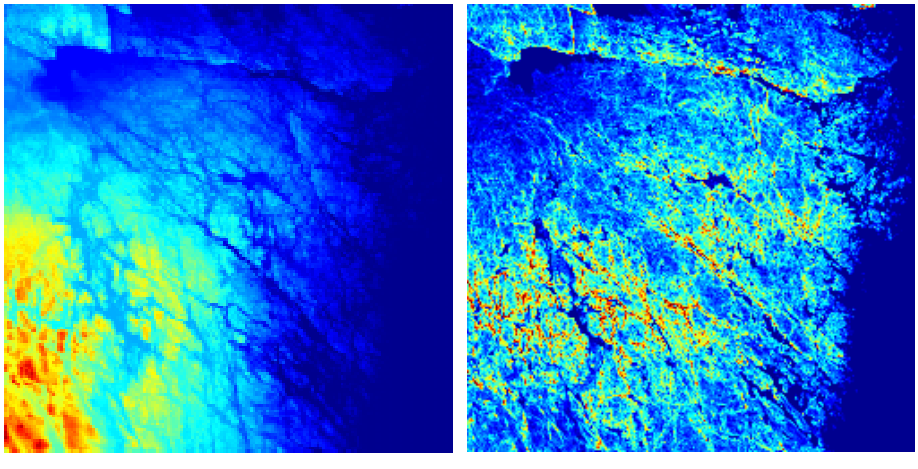


Figure 2.2: The left part shows the terrain height and the right part the information content in the map over a central part of Sweden.

by 100 km area of central Sweden. The pure terrain elevation samples are given in a uniform mesh of 50 by 50 meter resolution and shown to the left in the figure. Using interpolation from surrounding map values, the terrain map can be regarded as a known look-up table of terrain elevations as a function of position. Everything that in the real world cannot be found by interpolation from the database values must be regarded as noise. The right part of Figure 2.2 very clearly shows the lakes, and the coastline of the Baltic sea to the right in the map. The north-west part of the map is flat agricultural land with very little navigation information, while the southern part consists of very rough terrain with varying hills of some hundred meters height and narrow valleys that give a high information content. Maps such

as the right part of Figure 2.2 could be used for mission planning purposes, e.g., finding the most informative path to the final destination.

There are several commercial algorithms that solve the terrain navigation problem. Since the development has been driven by military interests, most of these are not very well documented in the literature. The most frequently referred algorithms for terrain navigation are TERCOM (terrain contour matching) and SITAN (Sandia inertial terrain-aided navigation). TERCOM is batch oriented and correlates gathered terrain elevation profiles with the map periodically [6, 75, 135]. The aircraft is not allowed to maneuver during data acquisition in TERCOM and therefore it has mainly been used for autonomous crafts, like cruise missiles. SITAN is recursive and uses a modified version of an extended Kalman filter (EKF) in its original formulation [91]. When flying over fairly flat or over very rough terrain, or when the aircraft is highly maneuverable, this algorithm does not in general perform well. In order to overcome these divergence problems parallel EKFs have been used in [31, 88]. Another widespread system is the TERPROM system, developed by British Aerospace, can be found in several NATO aircraft. It is a hybrid solution, in which an acquisition-mode correlates measurements in batch to find an initial position and in track-mode processes measurements recursively using Kalman filter techniques. However, due to commercial interests and because of its use as a classified military system, it is not as well documented in the literature as the previous two. One more recent and different approach that tries to deal with the nonlinear problems is VATAN [64]. In VATAN the Viterbi algorithm is applied to the terrain navigation problem, yielding a maximum a posteriori position estimate.

In this work, we take a completely statistical view on the problem and solve the matching with the map as a recursive nonlinear estimation problem. The conceptual solution is described in the following section and an approximate implementation in Section 2.3. Simulation results with this implementation are presented Section 2.4.

2.2 The Bayesian Approach to Terrain Navigation

As shown in Figure 2.1 the difference between the altitude estimate and the measured ground clearance yields a measurement of the terrain elevation. Assuming additive measurement noise the terrain elevation y_t relates to the current aircraft position x_t according to

$$y_t = h(x_t) + e_t \tag{2.2}$$

where the function $h(\cdot) : \mathbb{R}^2 \mapsto \mathbb{R}$ is the terrain elevation map. The measurement noise e_t is a white process with some known distribution $p_{e_t}(\cdot)$. This measurement error models both the errors in the radar altimeter measurements, the current altitude estimate and errors originating from the interpolation in the terrain map not perfectly resembling the real world. Let u_t denote the estimate of the relative movement of the aircraft between two measurements obtained from the INS.

Modeling the dead-reckoning drift of the INS with a white additive process v_t, the absolute movement of the aircraft obeys a simple linear relation

$$x_{t+1} = x_t + u_t + v_t \tag{2.3}$$

where v_t is distributed according to some assumed known probability density function $p_{v_t}(\cdot)$. Summarizing equations (2.2) and (2.3) yields the nonlinear model

$$\begin{aligned} x_{t+1} &= x_t + u_t + v_t \\ y_t &= h(x_t) + e_t \end{aligned} \qquad t = 0, 1 \ldots \tag{2.4}$$

where v_t and e_t are mutually independent white processes, both of them uncorrelated with the initial state x_0 which is distributed according to $p(x_0)$. One may argue that the INS estimate u_t should be regarded as a sensor measurement instead of as a known parameter in the state transition equation,

$$u_t = x_{t+1} - x_t + e_t^{\text{INS}}.$$

This would require the introduction of a new state vector incorporating both x_t and x_{t+1} in order to retain the Markovian property of the state space model. Instead, we choose to include the error e_t^{INS} in the process noise v_t in (2.4). This limits the state dimension and drastically reduces the computational power required to recursively compute an approximation of the conditional density of the states.

The objective of the terrain navigation algorithm is to estimate the current aircraft position x_t using the observations collected until present time

$$\mathbb{Y}_t = \{y_i\}_{i=0}^t.$$

With a Bayesian approach to recursive filtering, everything worth knowing about the state at time t is condensed in the conditional density $p(x_t \,|\, \mathbb{Y}_t)$. With some abuse of notation, the distribution of a generic random variable z conditioned on another related random variable w is

$$p(z \,|\, w) = \frac{p(z\,,\, w)}{p(w)} = \frac{p(w \,|\, z)\, p(z)}{p(w)} \tag{2.5}$$

Assume that $p(x_t \,|\, \mathbb{Y}_{t-1})$ is known and apply (2.5) to the last member in the set $\mathbb{Y}_t$,

$$p(x_t \,|\, \mathbb{Y}_t) = \frac{p(y_t \,|\, x_t\,,\, \mathbb{Y}_{t-1})\, p(x_t \,|\, \mathbb{Y}_{t-1})}{p(y_t \,|\, \mathbb{Y}_{t-1})}.$$

Inserting the model (2.4) and noting that the denominator is a scalar normalization constant yield,

$$\begin{aligned} p(x_t \,|\, \mathbb{Y}_t) &= \alpha_t^{-1}\, p_{e_t}(y_t - h(x_t))\, p(x_t \,|\, \mathbb{Y}_{t-1}) \\ \alpha_t &= \int p_{e_t}(y_t - h(x_t)) p(x_t \,|\, \mathbb{Y}_{t-1})\, dx_t \end{aligned}$$

which describes the influence of the measurement. Using (2.5) the joint density of the states at two measurement instants is

$$p(x_{t+1}\,,\,x_t) = p(x_{t+1} \,|\, x_t)\, p(x_t).$$

The density update between two measurements is found by marginalizing this expression on the state x_t and inserting (2.4),

$$p(x_{t+1} \,|\, \mathbb{Y}_t) = \int p_{v_t}(x_{t+1} - x_t - u_t) p(x_t \,|\, \mathbb{Y}_t)\, dx_t.$$

This completes one iteration of the recursive solution. Summarizing the derivation, the Bayesian formula for updating the conditional density is initiated by $p(x_0 \,|\, \mathbb{Y}_{-1}) = p(x_0)$ and calculated as

$$\begin{aligned} p(x_t \,|\, \mathbb{Y}_t) &= \alpha_t^{-1}\, p_{e_t}(y_t - h(x_t))\, p(x_t \,|\, \mathbb{Y}_{t-1}) \\ p(x_{t+1} \,|\, \mathbb{Y}_t) &= \int p_{v_t}(x_{t+1} - x_t - u_t) p(x_t \,|\, \mathbb{Y}_t)\, dx_t \end{aligned} \tag{2.6}$$

where

$$\alpha_t = \int p_{e_t}(y_t - h(x_t)) p(x_t \,|\, \mathbb{Y}_{t-1})\, dx_t.$$

The Bayesian solution is a density function describing the distribution of the states given the collected measurements. From the conditional density, a point estimate such as the minimum mean square error estimate can be formed

$$\hat{x}_t = \int x_t\, p(x_t \,|\, \mathbb{Y}_t)\, dx_t. \tag{2.7}$$

Assuming that this estimate is unbiased, the covariance

$$C_t = \int (x_t - \hat{x}_t)(x_t - \hat{x}_t)^T\, p(x_t \,|\, \mathbb{Y}_t)\, dx_t \tag{2.8}$$

quantizes the accuracy of the estimate. Equation (2.8) is convenient when comparing (2.7) with estimates from other navigation systems.

The recursive update of the conditional density (2.6) describes how the measurement y_t and the relative movement u_t affect the knowledge about the aircraft position. With each new terrain elevation measurement, the prior distribution $p(x_t \,|\, \mathbb{Y}_{t-1})$ is multiplicatively amplified by the likelihood of the measurement y_t. This means that the conditional probability will decrease in unlikely areas and increase in areas where it is likely that the measurement was obtained. Between two measurements, the density function $p(x_t \,|\, \mathbb{Y}_t)$ is translated according to the relative movement of the aircraft obtained from the INS and convolved with the density function of the error of this estimate. Thus the support and shape of the conditional density will adapt to areas which fit the measurements well and follow

the movement obtained from the INS. It is worth noting that (2.6) is the Bayesian solution to (2.4) for all possible nonlinear functions $h(\cdot)$ and for any noise distributions $p_{v_t}(\cdot)$ and $p_{e_t}(\cdot)$. In the special case of linear measurement equation and Gaussian distributed noises the equations above coincide with the Kalman filter [2].

Computationally, each iteration of the Bayesian solution (2.6) consists of solving several integrals. Due to the unstructured nonlinearity $h(\cdot)$, these integrations are in general impossible to solve in closed form and therefore there exists no solution that updates the conditional density analytically. The implementation must therefore inevitably be approximate. A straightforward way to implement the solution is to simply evaluate the recursion in several positions inside the area where the aircraft is assumed to be and update these values further through the recursion. With such a quantization of the state space, the integrals in (2.6) turn into sums over the chosen point values. The earliest reference of such a numerical approach to solving the nonlinear filtering problem is [34]. More recent references involve the p-vector approach in [137] and a slightly different approach, presented in [103], using a piecewise constant approximation to the density function. In the terrain navigation problem the state dimension is two and the quantization can in general be viewed as a bed-of-nails where the length of each nail corresponds to a certain elementary mass in that position. The implementation described in this paper is therefore labeled the point-mass filter (PMF).

2.3 The Point-Mass Filter

Assume that N grid points in $\mathbb{R}^2$ have been chosen for the approximation of $p(x_t \,|\, \mathbb{Y}_t)$. Introduce the notation

$$x_t(k) \qquad k = 1, 2, \ldots, N$$

for these N vectors in $\mathbb{R}^2$. Each of these N grid points has a corresponding probability mass

$$p(x_t(k) \,|\, \mathbb{Y}_t) \qquad k = 1, 2, \ldots, N.$$

In order to obtain a simple and efficient algorithm, the grid points are chosen from a uniform rectangular mesh with resolution of δ meters between each grid point. Each integral operation in (2.6) is approximated by a finite sum over the grid points with nonzero weight

$$\int_{\mathbb{R}^2} f(x_t)\, dx_t \approx \sum_{k=1}^{N} f(x_t(k))\, \delta^2.$$

Applying this approximation to (2.6) yields the Bayesian point-mass recursion:

$$\begin{aligned} p(x_t(k) \,|\, \mathbb{Y}_t) &= \alpha_t^{-1} p_{e_t}(y_t - h(x_t(k)))\, p(x_t(k) \,|\, \mathbb{Y}_{t-1}) \\ x_{t+1}(k) &= x_t(k) + u_t \quad k = 1, 2, \ldots, N \\ p(x_{t+1}(k) \,|\, \mathbb{Y}_t) &= \sum_{n=1}^{N} p_{v_t}(x_{t+1}(k) - x_t(n))\, p(x_t(n) \,|\, \mathbb{Y}_t)\ \delta^2 \end{aligned} \tag{2.9}$$

where

$$\alpha_t = \sum_{k=1}^{N} p_{e_t}(y_t - h(x_t(k)))\, p(x_t(k) \,|\, \mathbb{Y}_{t-1})\ \delta^2. \tag{2.10}$$

The time update has been split into two parts. First the grid points are translated with the INS relative movement estimate u_t and then the probability mass density is convolved with the density $p_{v_t}(\cdot)$. The point estimate (2.7) is computed at each iteration as the center of mass of the point-mass density,

$$\hat{x}_t = \sum_{k=1}^{N} x_t(k) p(x_t(k) \,|\, \mathbb{Y}_t)\ \delta^2.$$

Hence, the estimate does not necessarily fall on a grid point.

In order to follow the aircraft movements the grid must be adapted to the support of the conditional density. After each measurement update, every grid point with a weight less than $\varepsilon > 0$ times the average mass value

$$\frac{1}{N} \sum_{k=1}^{N} p(x_t(k) \,|\, \mathbb{Y}_t) = \frac{1}{N\delta^2}.$$

is removed from the grid. The new set of grid points is defined by

$$\left\{ x_t(k) : p(x_t(k) \,|\, \mathbb{Y}_t) > \varepsilon / N\delta^2 \right\}.$$

The weights need to be re-normalized after this truncation operation. The truncation will make the algorithm focus on areas with high probability and remove grid points in areas where the conditional density is small. The basic grid resolution δ will however not be affected by the truncation. When the algorithm is initialized, the uncertainty about the aircraft position is usually rather high. The prior will then have a large support, and naturally it is not interesting to have a high grid resolution. Instead we start with a sparse grid and run the algorithm and remove weights using the truncation operation above until the number of remaining grid points falls below some threshold N_0. Then the mesh resolution can be increased and the algorithm continued to process new measurements, updating the conditional density in the new dense grid. The up-sampling is performed by placing one grid point between every neighboring grid point in the mesh using linear interpolation to determine its weight. This will yield a doubling of the mesh resolution.

The convolution in (2.9) will introduce some extra grid points along the border of the point-mass approximation, increasing the support of the mesh. If the measurements have low information content there will be a net-increase of grid points even though some are removed by the truncation operation. Therefore, if the number of grid points increases above some threshold N_1, the mesh is decimated by removing every second grid point from the mesh, halving the mesh resolution.

Hence, the number of grid points N, the point-mass support and the mesh resolution δ is automatically adjusted through each iteration of the algorithm using the design parameters ε, N_0 and N_1. An illustration is given in Figure 2.6. In summary the PMF algorithm consists of (2.9) and the appropriate resampling of the grid described above. Details about the implementation and the grid refinement procedure can be found in [16].

The refinement of the grid support and resolution in the PMF described above is of course *ad hoc*, and one may wonder if this actually works. The Cramér-Rao lower bound is a fundamental limit on the achievable algorithm performance which can be used to evaluate the average performance of the PMF and verify that the filter solves the nonlinear filtering problem with near optimal performance. The results are presented here without proofs or derivations, see [16, 17, 23] for details. Let $\mathrm{N}(x; m, P)$ denote the n-dimensional Gaussian distribution with mean vector μ and covariance matrix P

$$\mathrm{N}(x; \mu, P) = \frac{1}{\sqrt{(2\pi)^n |P|}} \exp\left(-\tfrac{1}{2}(x-\mu)^T P^{-1}(x-\mu)\right).$$

Inserting Gaussian distributions in (2.4),

$$p_{e_t}(e_t) = \mathrm{N}(e_t; 0, R_t) \qquad p_{v_t}(v_t) = \mathrm{N}(v_t; 0, Q_t) \qquad p(x_0) = \mathrm{N}(x_0; \hat{x}_0, P_0),$$

the Cramér-Rao lower bound for the one step ahead prediction of the states satisfies the matrix (Riccati) recursion,

$$P_{t+1} = P_t - P_t H_t (H_t^T P_t H_t + R_t)^{-1} H_t^T P_t + Q_t$$

initiated with P_0. Above H_t is the gradient of $h(\cdot)$ evaluated at the true state value at time t,

$$H_t = \nabla h(x_t).$$

Thus, the Cramér-Rao bound is a function of the noise levels and the gradient of the terrain along the true state sequence.

The Cramér-Rao bound sets a lower limit on the estimation error covariance which depends on the statistical properties of the model (2.4) and on the algorithm used. A Monte Carlo simulation study is performed to determine the average performance of the algorithm for comparison with the Cramér-Rao bound. The Root Mean Square (RMS) Monte Carlo error for each fixed time instant is lower bounded by the Cramér-Rao bound,

$$\sqrt{\frac{1}{M}\sum_{i=1}^{M} \|x_t - \hat{x}_t^i\|^2} \gtrsim \sqrt{\operatorname{tr} P_t}$$

where $\hat{x}_t^i$ is the one step ahead prediction of the states at time t in Monte Carlo run i.

Figure 2.3 shows a 300 samples long track over a part of the terrain map from Figure 2.2. The aircraft travels from right to left. The Cramér-Rao bound and the

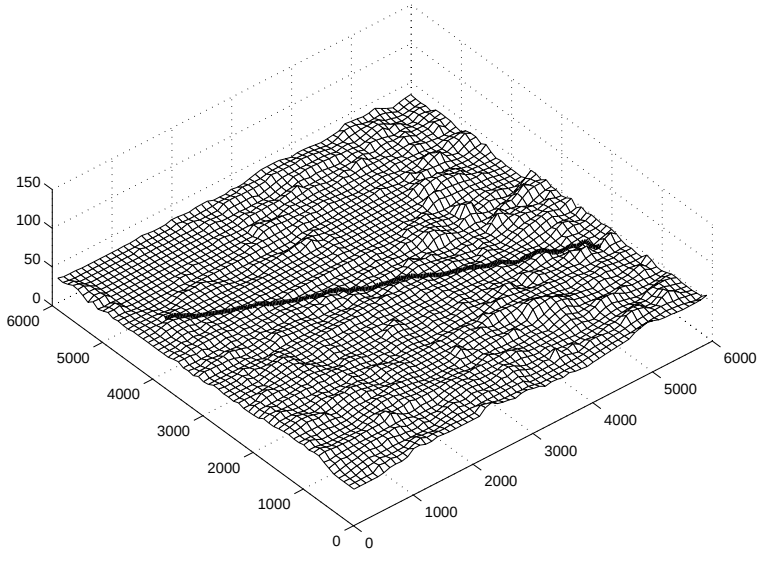


Figure 2.3: The simulation area and the true track, axes are labeled in meters.

obtained RMS error from 1000 Monte Carlo simulations with the PMF is shown in Figure 2.4. The PMF is initiated with a prior with large support using a low resolution of the grid, this yields far from optimal performance during the first half of the simulations. However, as the grid resolution increases the RMS error decreases and when the PMF resolution reaches a steady state level, the filter performance reaches the optimal bound. Note also that both the bound and the performance depend on the terrain variations along the true track. The Monte Carlo evaluation above shows that the grid refinement method used in the PMF works very well and that the filter achieves the optimal performance when the grid is chosen dense enough.

2.4 Simulation Evaluation

The terrain map used in these simulations is the same terrain database over a part of Sweden as is shown in Figure 2.2. A contour plot over this terrain map and the true simulated aircraft flight path is shown in Figure 2.5. The aircraft starts heading south, after a few turns over the rough part of the map it flies over a part of the Baltic sea and then turns back and completes the counter-clockwise

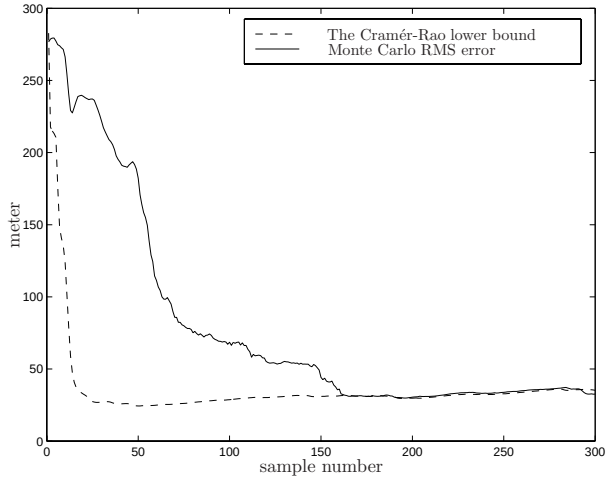


Figure 2.4: Monte Carlo root mean square error compared with the Cramér-Rao bound.

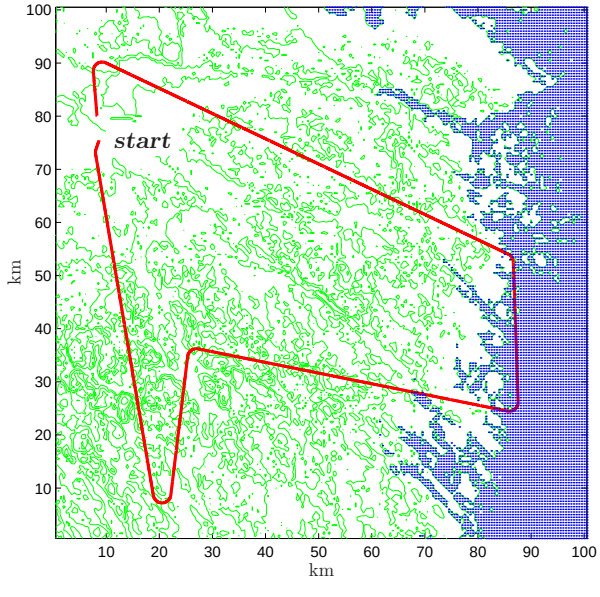


Figure 2.5: The simulation track over the terrain database.

lap. The simulated aircraft track, the INS measurements and the radar altimeter measurements have all been generated in an advanced realistic simulator used by the navigation systems development department at Saab Dynamics. The track has a duration of 25 minutes and is sampled at a rate of 10 Hz. The aircraft has an average speed of Mach 0.55, and the manoeuvres are simulated as coordinated turns.

The INS position estimate $\hat{x}_0$ is initiated with an error of 1000 m in both the north and the east direction. The prior density is chosen as a Gaussian distribution centered at the erroneous INS estimate

$$p(x_0) = \mathrm{N}(x_0; \hat{x}_0, 1000^2 I_2). \tag{2.11}$$

The initial grid resolution used in the PMF to sample this function is $\delta = 200$ m. The dead-reckoning drift in the INS is simulated as a constant bias of 1 m/s in each channel. The distribution used in the algorithm to model this drift is Gaussian $p_{v_t}(v_t) = \mathrm{N}(v_t; 0, 4I_2)$. The choice of Gaussian distributions has proven successful in the simulations but any other suitable distribution that better models the position drift and the initial uncertainty may be used. There is no restriction in the PMF to Gaussian noises: the only assumption is that the noise can be regarded as white.

Different sensor models are used when generating the simulated measurements. Depending on the terrain category beneath the aircraft at the measuring instant, both the bias and the variance of the radar altimeter are adjusted. For example, flying over dense forest the radar altimeter has a bias of 19 m with a large variance. Additional noise is added to the measurement to simulate that the radar altimeter measurement performance degrades with increasing ground clearance distance. The density used to capture these effects in the PMF algorithm is a mixture of two Gaussian distributions,

$$p_{e_t}(e_t) = 0.8\ \mathrm{N}(e_t; 0, 2) + 0.2\ \mathrm{N}(e_t; 15, 9).$$

This choice can be interpreted as on the average every fifth measurement being biased due to reflection in trees or buildings. The truncation and resampling parameters used in the PMF are,

$$\varepsilon = 10^{-3}, \qquad N_0 = 1000, \qquad N_1 = 5000.$$

The simulation result from the first three recursions is depicted in Figure 2.6. Starting with the Gaussian prior (2.11), the first measurement amplifies the probability in several regions and removes samples of low probability. After the second recursion, the grid resolution is increased to 100 m and the third recursion removes even more samples and a single peak of the density shows the most probable aircraft position while the uncertainty still is rather large. The bounding box indicating the support of the prior is shown as a comparison with the support of each of the filter densities. The irregular shapes of these densities shows how the unstructured nonlinear terrain gives a filter density which is hard to approximate with smooth

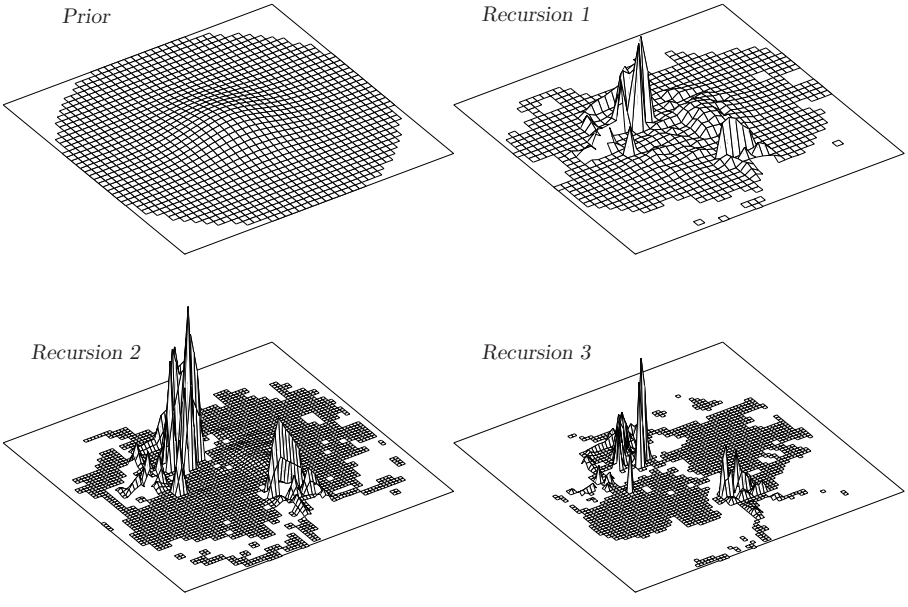


Figure 2.6: The first three recursions of the algorithm

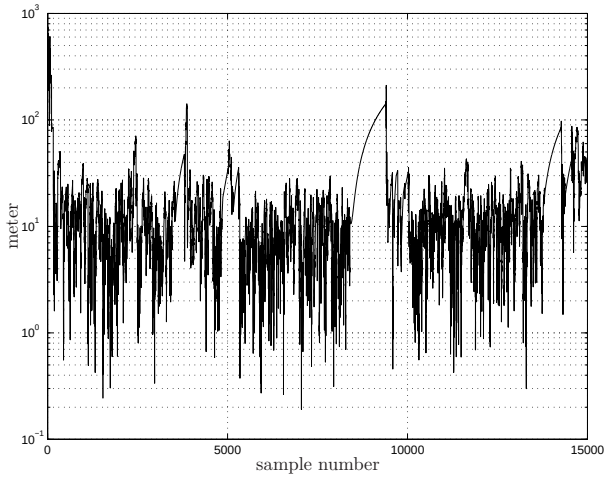


Figure 2.7: Estimation error along the simulation track, in logarithmic scale.

functions or local linearizations. Figure 2.7 shows the estimation error along the simulation track. Here it is obvious that the performance depends on the covered terrain. The error converges rapidly from the initial error of more than 1 km down to an error less than 30 m. When the aircraft reaches the Baltic sea the measurements have little information and the error increases with the drift of the INS. Once back over land, the estimate accuracy increases and a trend towards worse performance is visible when the aircraft covers the low informative areas of the map during the final part of the lap. The resolution of the grid is automatically adjusted and varies between 200 m and 0.78 m along the simulation track. A common navigation performance parameter is the circular error probable (CEP) which is the median of the position error. The simulation yields a median error of 12.2 m CEP. As a comparison, in [88] an error of 50 m CEP is reported and in [31] a value of 75 m is obtained. It should be remarked that both these values are found during field tests and not simulations.

2.5 Conclusion

The performance of terrain navigation depends on the size of the terrain gradient in the area. The point-mass filter described in this work yields an approximate Bayesian solution that is well suited for the unstructured nonlinear estimation problem in terrain navigation. It recursively propagates a density function of the aircraft position. The shape of the point-mass density reflects the estimate quality, this information is crucial in navigation applications where estimates from different sources often are fused in a central filter. The Monte Carlo simulations show that the approximation can reach the optimal performance and the realistic simulations in Section 2.4 show that the navigation performance is very high compared with other algorithms and that the point-mass filter solves the recursive estimation problem for all the types of terrain covered in the test.

The main advantages of the PMF is that it works for many kinds of nonlinearities and many kinds of noise and prior distributions. The mesh support and resolution are automatically adjusted and controlled using a few intuitive design parameters. The main disadvantage is that it cannot solve estimation problems of very high dimension since the computational complexity of the algorithm increases drastically with the dimension of the state space. The implementation used in this work shows real-time performance for two dimensional and in some cases three dimensional models, but higher state dimensions are usually intractable.

2.6 Acknowledgment

The work has been partly sponsored by ISIS, a NUTEK Competence Center at Linköping University. The simulation data and the terrain database have been provided by Saab Dynamics, Linköping Sweden. Valuable detailed explanations about the navigation application have been provided by several employees of Saab, Linköping.

3

Bayesian Estimation

Statistical estimation deals with the problem of inferring knowledge about parameters indirectly observable from the outcome of a related experiment. Usually, the experiment is a measurement or observation of a real world phenomenon, and the parameter is a physical quantity that affects the measurement in a known manner. In recursive estimation, the inferred knowledge about the parameters is updated continuously as new measurements are collected. This recursive processing of observations is suitable in problems where the parameters have dynamic properties that make them change with time, or when the application demands estimates with certain frequency based on the sequence of measurements observed so far. We label the estimation problem *recursive* when there is a demand for such recursive processing of the observations. With the Bayesian view on estimation, both the sought parameters and the observations are stochastic entities. This fundamental paradigm yields a unifying framework for estimation problems where the inference result is a conditional density function for the parameters given the observational outcome.

This chapter is a review of basic estimation theory and it serves as a theoretical platform for the sequel of the thesis. There are several excellent textbooks that give a much deeper and more thorough presentation of estimation theory than found herein. The book of Papoulis [120] presents the theory of probability and random processes from first principles. This reference also contains some interesting historical notes on the development of probability theory. A brief review of the

foundational concepts from probability theory is provided in the appendix to this chapter. The text in the appendix is based on [120]. Real valued parameters and observations are considered throughout this chapter. A thorough measure theoretic presentation of probability theory is given in the monograph by Chung [40].

The review of estimation theory presented in this chapter is mainly based on the classical book of Van Trees [151]. Both the parametric and Bayesian approaches to estimation are given by Van Trees, and the connections between estimation and detection are strongly emphasized. A complement to the rigorous book by Van Trees [151] that covers similar subjects in less detail is given by Scharf [130]. A rather theoretical, and purely Bayesian decision theoretic presentation of statistical inference is given in the monograph of Robert [127]. This reference contains a strong argumentation in favor of the Bayesian paradigm in general. On the other side, the pure parametric viewpoint is given by Lehmann [109], who focuses on classical inference where point estimators and their theoretical properties are presented in great depth.

Jazwinski [94] treats the recursive estimation problem both in continuous and discrete time, and for linear and nonlinear models. A more detailed presentation of linear recursive estimation is given by Anderson and Moore [2], where only discrete time is considered. The manuscript of Kailath et al. [96] will probably be the key reference to linear estimation when it finally goes into print.

3.1 Notational Conventions

A review of probability theory from the axiomatic definition to the concept of conditional probability density functions is given in Appendix 3.A. This section gives a very brief summary of the notational conventions, described in more detail in Appendix 3.A.

Unless stated differently, x denotes a generic n-dimensional random parameter vector and y a p-dimensional observation vector. Commonly in the literature, the word parameter is saved for nonrandom fixed entities while random entities are labeled states. In the following section, we make no linguistical distinction between random and nonrandom entities, all sought entities are labeled parameters regardless of if they are equipped with a prior distribution or not. Random variables are sometimes typed in bold face, $\boldsymbol{x}$, when it serves to clarify the underlying meaning. Generally, $p(x)$ denotes the probability density function for the random variable given in the argument, in this case $\boldsymbol{x}$. When this rule does not apply, the random variable is indicated by a subscript, i.e., $p(y)\big|_{y=x} = p_y(x)$. Likewise, mathematical expectation is performed w.r.t. all random variables given in the argument unless a subscript indicates which variable should be affected by the integration, e.g., $\mathsf{E}(f(x)) = \mathsf{E}_x(f(x))$.

The exact meaning of the conditional densities $p(x \,|\, y)$ and $p(y \,|\, x)$ is defined in Appendix 3.A. Expectation w.r.t. a conditional density is denoted $\mathsf{E}(x \,|\, y)$, and it follows the same subindex rule as regular expectation, e.g., $\mathsf{E}(f(x,y)) = \mathsf{E}_y(\mathsf{E}(f(x,y) \,|\, y)) = \mathsf{E}_y(\mathsf{E}_{x|y}(f(x,y)))$.

3.2 Bayesian Estimation

The objective of the estimation procedure is to gather information about the value of the parameter x given an observation of an experimental outcome, y. In statistical estimation it is customary to treat the observation y as a random vector, often justified by assumptions about random measurement noise, imprecise measurement equipment or other unmodeled effects that may lend themselves for random modeling. The observation vector is assumed to have a probability density function belonging to a class indexed by the parameters, $p(y \mid x)$. Hence, if the true parameter value was known, the complete statistical properties of the measurement would also be known. In a Bayesian framework the parameter vector itself is also referred to as a random vector. The random vector x is assumed having a known *prior density function* $p(x)$. This prior distribution encompasses everything known, and unknown, about the parameters prior to observing the experimental outcome.

The result of the experiment is a realization of the random variable $\boldsymbol{y}$. Observing the event $\{\boldsymbol{y} = y\}$ yields, by means of Bayes' law (3.A.24), that the knowledge about the parameters is altered so that

$$p(x \mid y) = \frac{p(y \mid x)p(x)}{p(y)}. \tag{3.1}$$

The meaning of this relation is described in detail in Appendix 3.A. With a purely Bayesian view on the estimation problem the *posterior density function* $p(x \mid y)$ describes everything about the parameter x after the experimental outcome has been observed. With the experiment result y at hand, the denominator of (3.1) is just a scalar positive constant which can be found by marginalization,

$$p(y) = \int_{\mathbb{R}^n} p(y \mid x)p(x)\, dx.$$

Hence, in order to compute (3.1) one only needs to specify the product $p(y \mid x)p(x)$.

Proposition 3.1
A Bayesian estimation problem is defined by the joint density of the parameters and the observations, $p(x, y) = p(y \mid x)p(x)$.

From a Bayesian viewpoint, the likelihood and the prior, or alternatively the joint density, define the statistical model for the estimation problem, while the posterior (3.1) is its solution. Once the information $\{\boldsymbol{y} = y\}$ is at hand, the parameters, priorly regarded as the random variable $\boldsymbol{x}$, should now be regarded as the random variable $\boldsymbol{x} \mid y$. The posterior density can be used to deduce the probability of any characteristic of the parameter given the data. Hence, the posterior density should always be regarded as the most general solution to the estimation problem even if the primary goal of the inference often is aimed at certain features of the posterior.

3.2.1 Estimates

The posterior is a general, but rather complex answer to the inference problem. Each candidate parameter value in $\mathbb{R}^n$ yields a value of $p(x \mid y)$ reflecting the posterior probability of that parameter value. An *estimate* $\hat{x}$ is an educated guess of the parameter value given the observations at hand. It is often more desirable to determine an estimate than the complete posterior density. Each realization of the measurement yields a new estimate, and generally the rule of passing from an observation to a parameter estimate is a function termed an *estimator*. Hence, an estimator is a function from the observation space to the parameter space $\hat{x} : \mathbb{R}^p \to \mathbb{R}^n$. The argument of this function is explicitly indicated, $\hat{x}(y)$, only when the dependency needs to be clarified. Note that the estimator should be regarded as a random variable, since it is a function of the random variable $\boldsymbol{y}$, while the estimate is a deterministic vector since it depends on the observed realization y.

A natural way to find suitable estimators is to define a penalty for choosing an erroneous estimate. Let $L(\hat{x}(y), x)$ denote a cost function, reflecting a user defined penalty for erroneous estimates $\hat{x} \neq x$. In general, $L(\hat{x}, x)$ is defined such that the greater the discrepancy $x - \hat{x}$, the larger the cost. Without loss of generality, we can assume that the cost function is positive and that $L(x, x) = 0$ is its unique minimum. The Bayesian risk $\mathfrak{R}$ associated with an estimator $\hat{x}$ is defined as the expected cost,

$$\mathfrak{R} \triangleq \mathsf{E}(L(\hat{x}(y), x)) \,,$$

where expectation is over both x and y. The optimal choice of $\hat{x}(y)$, using a cost defined by $L(\hat{x}(y), x)$, is the one that minimizes the Bayesian risk,

$$\hat{x}(y) = \arg\min_{x^\star(y)} \int_{\mathbb{R}^{n+p}} L(x^\star(y), x) p(x, y) \, dx \, dy,$$

where the minimization is over all possible functions $x^\star(y)$. Inserting $p(x, y) = p(x \mid y)p(y)$ splits the integration into two parts

$$\hat{x}(y) = \arg\min_{x^\star(y)} \int_{\mathbb{R}^p} \int_{\mathbb{R}^n} L(x^\star(y), x) p(x \mid y) \, dx \; p(y) \, dy.$$

Since both $L(x^\star, x)$ and $p(x \mid y)$ are positive, the inner integral is positive given any y. Furthermore, since $p(y)$ also is positive, the value of $x^\star(y)$ that minimizes the risk is the value that minimizes the inner integral,

$$\hat{x}(y) = \arg\min_{x^\star(y)} \int_{\mathbb{R}^n} L(x^\star(y), x) p(x \mid y) \, dx \tag{3.2}$$

for each y. The optimization problem (3.2) defines the Bayesian estimation problem and each choice of cost function gives different estimators. As an alternative

to (3.2), a minimax approach can be used. Then, the estimate that minimizes the maximal cost is chosen,

$$\hat{x}(y) = \arg\min_{x^\star(y)} \max_{x} L(x^\star(y), x). \tag{3.3}$$

In estimation problems it is often the case that the cost function only depends on the estimation error

$$\tilde{x} \triangleq x - \hat{x},$$

and the abbreviated notation $L(x - \hat{x})$ is used. This is no general consideration though, there may, e.g., be situations where the cost of erroneously choosing certain estimates should be extra high since those estimates are used to take some crucial actions. This is exemplified at the end of this section. Nevertheless, the cost function $L(\tilde{x})$ is rather general and can, e.g., handle cases when it is a much more severe error to overestimate a quantity than to underestimate it. To illustrate the use of cost functions to determine estimates, two standard choices of cost function and their solutions to (3.2) follow below, see [151] for details.

First, we introduce a compact notation for quadratic norms. Let $\|x\|_A^2 \triangleq x^T A x$ for any square matrix A and compatible matrix or vector x, and introduce the abbreviated notation $\|x\|^2 = \|x\|_I^2$. Moreover, let $A > 0$ denote that A is symmetric and positive definite. The, possibly weighted, mean-square error cost is defined by

$$L_{\text{MS}}(x - x^\star) \triangleq \|x - x^\star\|_Q^2 \tag{3.4}$$

where $Q > 0$ is a weighting matrix. The minimum mean-square error estimate is thus defined by

$$\hat{x}_{\text{MS}} \triangleq \arg\min_{x^\star} \int_{\mathbb{R}^n} \|x - x^\star\|_Q^2 \, p(x \,|\, y) \, dx.$$

Taking the gradient of the RHS integral w.r.t. $x^\star$ and setting the result equal to zero yield

$$-2 \int_{\mathbb{R}^n} Q x \, p(x \,|\, y) \, dx + 2 \int_{\mathbb{R}^n} Q x^\star \, p(x \,|\, y) \, dx = 0 \tag{3.5}$$

which is an equation for the unique minimum since the second derivative equals $2Q > 0$. Solving for $x^\star$ yields that the optimal estimate in the mean-square sense is

$$\hat{x}_{\text{MS}} = \int_{\mathbb{R}^n} x p(x \,|\, y) \, dx. \tag{3.6}$$

For obvious reasons, this estimate is also referred to as the conditional mean estimate.

Another common choice of cost function is the one that penalizes all errors equally,

$$L_{\text{UNIF}}(x - x^\star) \triangleq \begin{cases} 1 & \text{if } \|x - x^\star\|^2 \leq \frac{\Delta}{2} \\ 0 & \text{if } \|x - x^\star\|^2 > \frac{\Delta}{2} \end{cases}$$

given some small $\Delta > 0$. Then, as $\Delta \to 0$, the optimal estimate is the *maximum a posteriori* estimate,

$$\hat{x}_{\text{MAP}} \triangleq \arg\max_x p(x \,|\, y), \tag{3.7}$$

which will coincide with the maximum likelihood estimate when an uninformative prior $p(x)$ is used. For a derivation of (3.7) see Van Trees [151].

The cost functions given above are the two most common choices for Bayesian point estimation. Some other examples of cost functions are given in [94, 151]. One reason for the frequent use of these cost functions is the explicit form of the estimates (3.6) and (3.7), while for a generally chosen cost function, the estimate is only implicitly defined through (3.2). Another reason for choosing the conditional mean estimate is that it actually solves the general optimization problem (3.2) for a rather large class of cost functions, illustrated in the following two lemmas.

Lemma 3.1 (Property 1 of [151, p. 60])
Let the cost function $L(\tilde{x})$ be a symmetric convex function, and assume that the posterior density is symmetric about its mean. Then, the optimal Bayesian estimator (3.2) *is the conditional mean* (3.6).

Lemma 3.2 (Property 2 of [151, p. 61])
Let the cost function $L(\tilde{x})$ be symmetric and nondecreasing, and assume that the posterior density is symmetric about its mean, unimodal, and that it satisfies

$$\lim_{x \to \infty} L(x) p(x \,|\, y) = 0.$$

Then, the optimal Bayesian estimator (3.2) *is the conditional mean* (3.6).

Various interpretations of these two lemmas, together with proofs, can be found in [2, 94, 151]. Throughout this thesis we will only consider conditional mean and maximum a posteriori estimates.

The general framework of defining estimates through cost functions also applies to hypothesis testing and decision problems, as illuminated by the following example.

Example 3.1 (Bayesian Detection Problem)
Consider the problem of detecting a nonzero parameter vector x given a related observation y. This problem is a binary hypothesis test with

$$\begin{aligned} H_0 &: \quad x = 0 \\ H_1 &: \quad x \neq 0 \end{aligned}$$

The estimate $\hat{x}$ is the output of the detector, thus it either accepts or rejects the hypothesis H_0. The possible costs are defined by a (2×2) matrix

$$\begin{array}{c|cc} L(\hat{x}, x) & x = 0 & x \neq 0 \\ \hline \hat{x} : \textit{choose } H_0 & C_{00} & C_{01} \\ \hat{x} : \textit{choose } H_1 & C_{10} & C_{11} \end{array} \tag{3.8}$$

With the prior density

$$p(x) = \begin{cases} P_0 & \text{if } x = 0. \\ P_1 & \text{if } x \neq 0. \end{cases}$$

inserted into (3.2) *this will eventually lead to the well known* likelihood ratio test *[130, 151]*

$$\frac{p(y \,|\, x \neq 0)}{p(y \,|\, x = 0)} \underset{H_0}{\overset{H_1}{\gtrless}} \frac{P_0(C_{10} - C_{00})}{P_1(C_{01} - C_{11})}.$$

Hence, $\hat{x}$ is chosen to H_1 if the LHS is greater than the RHS, otherwise H_0 is chosen.

The likelihood ratio test is obtained when choosing the decision that minimizes the Bayesian risk (3.2), while the criterion (3.3) will lead to minimax type decisions. Detailed presentations of these concepts with deeper analogies between estimation and detection problems are carried out by Van Trees [151] and by Scharf [130]. In (3.8) one usually distinguishes the cost for false alarm, C_{10}, from that of missed detection, C_{01}. Thus, this is an example of a cost function which has different costs depending on both $\hat{x}$ and x, while for the estimates derived previously the cost function was a function only of the estimation error $L(\tilde{x})$.

3.3 Parametric Inference

With a so-called Fisherian, or parametric, view on estimation, the parameters are not regarded as random variables. Instead, the sought parameters are regarded as *fixed-but-unknown*, and the parameter value explicitly affects the statistical properties of the observation in a known manner. This connection between measurement and parameter is defined by a density function for the observation vector y parameterized by the parameter vector x, denoted $p(y \,|\, x)$. With a parametric view on estimation, $p(y \,|\, x)$ is regarded as a function of the parameters after inserting the measurement. Often, this *likelihood function* is written

$$l(x \,|\, y) \triangleq p(y \,|\, x)$$

to emphasize that it is regarded as a function of the first argument after inserting the observed y. Given the observed measurement y inserted into this function, the parameter vector x corresponding to the density most likely generating the

observed measurement is chosen. An estimate is thus formed by maximizing the likelihood function,

$$\hat{x}_{\mathrm{ML}} = \arg\max_x l(x \mid y)$$

this will lead to the theory of *maximum likelihood* estimation, rigorously presented in [109]. With a very flat, also called uninformative, prior $p(x)$, it is readily seen from (3.1) that the posterior will be almost proportional to the likelihood so that the Bayesian and parametric viewpoints in some sense coincide. Detailed presentations over similarities and discrepancies between Bayesian and parametric statistical estimation can be found in [33, 127].

It is sometimes argued that the demand for prior knowledge in order to define $p(x)$ makes the Bayesian viewpoint less attractive. This should therefore favor the parametric approach where no such knowledge is needed. Moreover, the spokesmen for the parametric view often claim that even if such prior knowledge exists it is hard to describe, at least in the form of a probability density function. Similar reasoning says that since the choice of prior often is rather subjective, it may seriously alter the resulting inference in a destructive way. These arguments speak in favor of the parametric viewpoint over the framework of Bayesian statistics. However, since both the likelihood and the prior define the inference problem according to Proposition 3.1, picking the correct likelihood reflects the same problem as picking a good prior.

In a practical situation we cannot guarantee that the real world observations that we base our inference on are exactly drawn from the probability model defined by the joint density $p(x, y)$. Likewise, there is no way to assure that the measurements we collect origin from a parameterized density function $p(y \mid x)$ given some x in the set of admissible parameters. Both these frameworks are merely models where statistics is used to describe a complex reality. If the reality was exactly described by, e.g., the model $p(x, y)$, the complete solution is given by the posterior density $p(x \mid y)$. If this is not the case, but we still base our inference and compute our estimates with respect to the density $p(x, y)$, it is an issue of robustness against modeling errors. Detailed discussions in this topic along with means to introduce robustness into Bayesian inference is given by Berger [14], Box and Tiao [33].

The need for defining a prior knowledge is not a demand but an option. Choosing an uninformative prior attenuates its effect on the inference result, while a prior with a well defined support will be clearly visible in the estimate. Hence, the Bayesian approach can be seen as a generalization of the parametric approach with an option to bias the result by choosing the support and shape of the prior density. Robert [127] delivers a strong argumentation in favor of the Bayesian viewpoint over the parametric one. General guidelines to the choice of prior distribution are given by Berger [14], Box and Tiao [33] and Robert [127].

3.4 Sensor Fusion

Conceptually, inference based on several sensors can be performed within either of the frameworks for statistical inference reviewed above. Since we allow for a vector valued observation, the measurement vector can straightforwardly be assembled from two physical measurements provided by different sensors. The statistical dependency between the two sensors is controlled by the choice of likelihood. Combining, e.g., two sensor outputs, y_1 and y_2, with independent measurement noise, a likelihood on the form

$$p(y \mid x) = p(y_1, y_2 \mid x) = p(y_1 \mid x)p(y_2 \mid x) \tag{3.9}$$

is a suitable choice. However, inference at this high level of generality is often not practically possible. Either the features that the sensors measure or the type of information they produce are usually so different that it is hard to model the behavior of all sensors in a complete framework of the type (3.9). Moreover, implementational constraints often induce that most processing of data need to be executed locally, at each sensor. This may put constraints on the inference that are hard to incorporate in the framework of (3.9).

The general problem of combining information regarding an object, or several objects, and the surrounding environment is commonly referred to as the *sensor fusion* problem. The key difference between traditional statistical inference and sensor fusion is that the information considered is of fundamentally different form and provided on different levels of abstraction. Surveys over the area are provided, e.g., by Waltz and Llinas [154], and Hall [82]. It is commonly assumed that the sensors are given in a decentralized architecture, Varshney [152] presents several issues regarding decentralized detection and estimation for sensor fusion.

The field of sensor, or information, fusion is fairly young, and has only recently been recognized as a separate research society. It has developed from a composition of several existing, rather different, branches of research. Issues raised in the field of sensor fusion concern conventional and non-conventional methods for statistical estimation and foundational concepts of probability. But sensor fusion also covers topics from computer science, artifical intelligence, image processing, physical modeling of the sensors and man-machine interface issues. In sensor management, the collected information is utilized to control the sensor, the inherent feedback leads to a risk of instability which can be analyzed by methods from the field of automatic control. Therefore, automatic control also has strong connections to sensor fusion applications.

It is our strong belief that the issue of combination of sensor measurements ultimately remains best handled within the sound framework of statistical inference. The basic concepts of probability and statistics presented in this chapter provide a solid foundation on which any sensor fusion methodology inevitably must rest. However, fusion of information of conceptually different type may very well call for unconventional methods, and in Section 3.4.3 we review such a novel framework for representation of sensor ambiguity that is commonly utilized in the sensor fusion literature. We emphasize that, although the Dempster–Shafer framework has been

considered as a generalization to Bayesian inference, this method has many things in common with the classical statistical paradigm of Bayesian inference. In the two subsections preceding the Dempster–Shafer survey, we present the general issue of combination of sensor measurements from a purely statistical point of view. The issues presented below serve as a foundation for the recursive estimation problem presented in Section 3.5. Recursive estimation is a type of sensor fusion since it is a sequential combination of measurements from one or several sensors.

3.4.1 Estimation Error Covariance

From a Bayesian perspective, the posterior density $p(x \mid y)$ describes everything worth knowing about the parameters after the experimental outcome has been observed. Even though the parameters are regarded as random, an estimate is often of more practical interest than the conditional density itself. An estimate computed from the posterior density is an educated guess of the vector parameter value. However, the estimate alone does not reveal anything about the relative goodness of this guess.

A convenient entity that quantifies the estimate quality is the mean square *error correlation matrix*

$$P \triangleq \mathsf{E}\big((x-\hat{x})(x-\hat{x})^T\big)\,, \tag{3.10}$$

if $\mathsf{E}(x-\hat{x}) = 0$ this quantity also coincides with the *estimation error covariance*. After observing the event $\{\boldsymbol{y} = y\}$ and computing a suitable estimate $\hat{x}(y)$ the error correlation matrix is straightforwardly computed from the posterior density

$$P = \int_{\mathbb{R}^n} (x-\hat{x})(x-\hat{x})^T p(x \mid y)\, dx.$$

Since the mean-square estimate (3.6) satisfies $\mathsf{E}(x - \hat{x}_{\mathrm{MS}} \mid y) = 0$, it has an error covariance matrix

$$P_{\mathrm{MS}} = \mathsf{Cov}\,(x - \hat{x}_{\mathrm{MS}} \mid y) = \mathsf{Cov}\,(x \mid y) \tag{3.11}$$

Hence, the first and second central moments of the posterior density $p(x \mid y)$ is the mean-square estimate and its estimation error covariance, respectively. Note that (3.11) gives the error covariance of the mean-square estimate based on the observed y and it quantifies the uncertainty about the this estimate. The error covariance of the estimator can be found by performing expectation over y as well, and will quantify the overall performance of the estimator, prior to collecting any observations. The estimator error covariance and estimator performance will be further discussed in Chapter 4.

3.4.2 Optimal Fusion of Estimates

In several applications there are different sensors observing related aspects of the same environment. Statistical inference is often performed at each sensor which

computes an estimate given the sensor data collected locally. The different estimates produced by the sensors are sent to a central fusion filter which has the role of combining the estimates in an optimal way. In such a scenario, a measure of the quality of each locally computed estimate is crucial for correct relative weighting in the central filter. Aircraft navigation is an application where system design often builds on such a distributed architecture. Several independent estimates of the aircraft kinematic state are delivered by navigation subsystems, often using statistical inference to interpret sensor measurements. Estimates and estimation error covariances are computed by each subsystem and integrated in a central navigation filter. Assuming that the different estimates have independent errors that are Gaussian distributed with zero mean and covariance given by the estimation error covariance, the mean-square estimate of the state given all distributed sensor information can be computed optimally by the central fusion filter. This fact relies on the fundamental property that the conditional density will be Gaussian in this case. Denote the generic n-dimensional Gaussian density with mean vector μ and covariance matrix P by

$$\mathrm{N}(x;\mu,P) \triangleq \frac{1}{\sqrt{(2\pi)^n|P|}} \exp\left(-\tfrac{1}{2}\|x-\mu\|^2_{P^{-1}}\right).$$

The following basic result is crucial for linear estimation, see [2] for details.

Theorem 3.1 (Fundamental Equations of Linear Estimation)
Let x and y be two jointly Gaussian vectors

$$p(x,y) = \mathrm{N}\left(\begin{bmatrix}x\\y\end{bmatrix}; \begin{bmatrix}\bar{x}\\\bar{y}\end{bmatrix}, \begin{bmatrix}P_{xx} & P_{xy}\\P_{yx} & P_{yy}\end{bmatrix}\right),$$

then

$$p(x\,|\,y) = \mathrm{N}\left(x; \bar{x} + P_{xy}P_{yy}^{-1}(y-\bar{y}), P_{xx} - P_{xy}P_{yy}^{-1}P_{yx}\right),$$

and thus $\hat{x}_{MS}(y) = \bar{x} + P_{xy}P_{yy}^{-1}(y-\bar{y})$, with an error covariance of $P_{xx} - P_{xy}P_{yy}^{-1}P_{yx}$.

Proof

$$p(x\,|\,y) = \frac{p(x,y)}{p(y)} \propto \sqrt{\frac{|P_{yy}|}{|\mathbb{P}|}} \exp\left(-\tfrac{1}{2}\|\left[\begin{smallmatrix}x\\y\end{smallmatrix}\right] - \left[\begin{smallmatrix}\bar{x}\\\bar{y}\end{smallmatrix}\right]\|^2_{\mathbb{P}^{-1}} + \tfrac{1}{2}\|y-\bar{y}\|^2_{P_{yy}^{-1}}\right) \tag{3.12}$$

where $\mathbb{P} = \left[\begin{smallmatrix}P_{xx} & P_{xy}\\P_{yx} & P_{yy}\end{smallmatrix}\right]$. Applying the transformation

$$\begin{bmatrix}P_{xx} & P_{xy}\\P_{yx} & P_{yy}\end{bmatrix} = \begin{bmatrix}I & P_{xy}P_{yy}^{-1}\\0 & I\end{bmatrix} \begin{bmatrix}P_{xx} - P_{xy}P_{yy}^{-1}P_{yx} & 0\\0 & P_{yy}\end{bmatrix} \begin{bmatrix}I & 0\\P_{yy}^{-1}P_{yx} & I\end{bmatrix}$$

and collecting terms in the exponent of (3.12) verifies the claim. □

With two estimates having independent, Gaussian, estimation errors, their mean-square optimal combination is given by the following widely applied theorem.

Theorem 3.2 (Sensor Fusion)
Let $\hat{x}_1$ and $\hat{x}_2$ be two estimates of x with independent Gaussian errors of covariance P_1 and P_2, respectively. Then, the mean-square estimate given the combined information is

$$\hat{x}_{FUSED} = (P_1^{-1} + P_2^{-1})^{-1}(P_1^{-1}\hat{x}_1 + P_2^{-1}\hat{x}_2) \tag{3.13}$$

and the resulting fused estimate will have an error covariance of

$$P_{FUSED} = (P_1^{-1} + P_2^{-1})^{-1}.$$

Proof With a Bayesian approach, two estimates cannot simply be combined without prior knowledge. Instead, we consider the first estimate as the prior information and the second estimate as an observation, and seek the posterior distribution of the parameters given all data. Given the first estimate, the distribution of the parameters is

$$\boldsymbol{x} \,|\, \hat{x}_1 = \hat{x}_1 + \tilde{\boldsymbol{x}}_1 \sim \mathcal{N}(\hat{x}_1, P_1)$$

where $\mathcal{N}(\mu, Q)$ denote a Gaussian distribution with mean vector μ and covariance P. The second estimate $\hat{x}_2 = x - \tilde{x}_2 = \hat{x}_1 + \tilde{x}_1 - \tilde{x}_2$, hence jointly

$$\begin{bmatrix} x \,|\, \hat{x}_1 \\ \hat{x}_2 \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \hat{x}_1 \\ \hat{x}_1 \end{bmatrix}, \begin{bmatrix} P_1 & P_1 \\ P_1 & P_1 + P_2 \end{bmatrix} \right).$$

Applying Theorem 3.1 yields

$$\begin{aligned} x \,|\, \hat{x}_1, \hat{x}_2 &\sim \mathcal{N}\left(\hat{x}_1 + P_1(P_1 + P_2)^{-1}(\hat{x}_2 - \hat{x}_1), P_1 - P_1(P_1 + P_2)^{-1}P_1\right) \\ &\sim \mathcal{N}\left((P_1^{-1} + P_2^{-1})^{-1}(P_1^{-1}\hat{x}_1 + P_2^{-1}\hat{x}_2), (P_1^{-1} + P_2^{-1})^{-1}\right) \end{aligned}$$

The claim follows from (3.6) and (3.11). □

The fusion formula of Theorem 3.2 is often used even if the errors involved are not Gaussian distributed, as might be the case in the aircraft navigation application. The results are often quite satisfactory anyway. However, one should always remember that in general, the estimate and estimate covariance cannot condense everything about the posterior density. The posterior density of a certain subsystem might, e.g., have several peaks of comparable size in different areas of the parameter space. Then, an estimate such as (3.6) might fall between these peaks in an area with very little probability. In such cases the formula (3.13) should be used with care.

3.4.3 Dempster-Shafer Combination

In this subsection we present a brief review over a method for statistical inference commonly utilized in sensor fusion applications. The method of Dempster–Shafer combination, or evidential reasoning, was originally developed in the statistical literature as a methodology for inference when the observation relation is set-valued. When there is a one-to-many transformation from the parameters of interest to the

space of observations it is not possible to deduce a unique conditional probability distribution for the state values. In a series of articles, Dempster [50, 51, 52] defines this problem and introduces the notion of upper and lower probabilities that can be determined from observations of this kind. Dempster claims this interval of probability being a natural generalization of standard Bayesian inference, and that it coincides with the latter method when the interval becomes trivially thin. The correctness of this assertion is greatly debated, e.g., in the discussion appended to [52]. Nevertheless, Shafer [132] later brought the ideas to the field of artifical intelligence and introduced several additional concepts defined on the solid statistical base of Dempster's work. When referring to the theory of evidential reasoning, or Dempster–Shafer theory, it is basically the framework described in [132] that is considered. Dempster's original work and the framework of Shafer both focus on discrete random variables from a countable set. In more recent work on this topic, Schneider [131] extends the theory to absolutely continuous distributions and connect it to the structural inference method of Fraser [68].

Dempster–Shafer theory considers a finite set of hypotheses $\Theta = \{H_i\}_{i=1}^{n}$ constituting the *frame of discernment* of the inference problem. A disjunction of hypotheses is referred to as a *proposition*, and the power set of all possible propositions is denoted 2^{Θ}. In classical inference, each singleton in Θ is assigned a posterior probability based on the observed measurement. The fundamental axioms of probability, given in Definition 3.A.1, yield that assigning a certain amount of probability to a hypothesis in Θ, the remaining probability mass must be distributed over the other hypotheses,

$$\Pr(H_1) = 1 - \Pr(\Theta \setminus \{H_1\}),$$

where $\Theta \setminus \{H_1\}$ is the complement of H_1 in Θ. In Dempster–Shafer theory this need is alleviated by letting each proposition be given a unique probability of its own, i.e., each subset of Θ is regarded as a distinct hypothesis. The yet unassigned probability mass is given as the probability for the certain proposition, Θ. A *probability mass distribution* assigns probabilities to the different propositions. Formally, this function $m : 2^{\Theta} \to [0, 1]$ needs to satisfy the axioms

$$0 \leq m(A) \leq 1 \qquad m(\emptyset) = 0 \qquad \sum_{B \in 2^{\Theta}} m(B) = 1$$

for all propositions $A \in 2^{\Theta}$. The combination of two independent mass distributions is given by Dempster's rule of combination, for each $A \in 2^{\Theta}$,

$$\begin{aligned} m(A) &= K \sum_{A_1 \cap A_2 = A} m_1(A_1) m_2(A_2) \\ K^{-1} &= 1 - \sum_{A_1 \cap A_2 = \emptyset} m_1(A_1) m_2(A_2). \end{aligned} \tag{3.14}$$

This formula distributes the probability to all propositions formed from disjunctions between the propositions available in the two original distributions $m_1(\cdot)$ and

$m_2(\cdot)$. A probability interval is computed from the combined probability mass. The *support* of a proposition A is defined as

$$\mathrm{Sp}(A) = \sum_{\{B:A\subseteq B\}} m(B),$$

which can be interpreted as all probability in $m(\cdot)$, in some sense pointing at A. The support is the upper limit of the probability interval for a proposition. The lower limit, known as the *plausibility* of the proposition, is defined as the lack of support for its negation, i.e., $\mathrm{Pl}(A) = 1 - \mathrm{Sp}(\neg A)$.

A justification of the inference rule (3.14) based on probability theory was presented in the original work of Dempster [51]. Generally, the combination rule (3.14) follows straightforwardly by regarding each proposition as a unique hypothesis and applying Bayes' rule. This connection, however, is not emphasized in work on Dempster–Shafer theory presented, e.g., in the senor fusion or artifical intelligence literature. It is even not emphasized in the original work of Dempster. Instead, initiated by the book of Shafer [132], a number of less rigorously verified extensions have been developed. Most of these are not theoretically justified from a statistical point of view.

The key feature of the Dempster–Shafer approach is the ability of introducing ignorance into the posterior probabilities. This effect is obtained by assigning probability to the certain event Θ. In classical inference, ignorance is handled by spreading the remaining probability equally over the possible hypotheses. The spokesmen of Dempster–Shafer theory claim that such an approach sometimes can yield misleading conclusions. Application of Dempster-Shafer theory is often directed towards identification of object properties, by utilizing observations on several levels of abstraction. Examples involve identification of aircraft type and class in target tracking systems [29, 35, 67] and object recognition in mobile robotic navigation [107]. The general conclusion drawn in all these references is that Dempster–Shafer not necessarily yields better performance than classical probabilistic inference. Even if the Dempster–Shafer framework might seem better suited for these problems, it is rather complex to implement and in recursive constellations it will often yield slower convergence towards a stationary result than classical inference [29, 35]. Arguments against Bayesian inference, on the basis of its demand for priors, have also been raised when comparing classical Bayesian inference with Dempster–Shafer type information fusion. However, it is rather easy to realize that both methods can utilize or ignore prior information as desired.

3.5 Recursive Estimation

Many applications of estimation theory tackle the problem of recursively determining, or tracking, parameters given measurements. Either the parameters admit some dynamical properties that make them change with time, and/or the estimate of the parameters is needed on-line demanding the information about the parameters to be updated with each new measurement. Examples of applications that fall

into this category are target tracking systems, used in both civilian and military aviation, and for naval applications. Here, the complete kinematic state consisting of position, velocity and possibly acceleration of the target is sought by passive or active position measurements. Other examples of recursive estimation are all types of navigation applications ranging from global aircraft and ship navigation down to autonomous robot navigation. Recursive estimation is also applied in adaptive control, where the parameters of a dynamical system working in closed loop are tracked using measurements of system input and output. One may also resort to recursive estimation in off-line applications with large amounts of data. In cases when the joint density $p(x, y)$ is very complicated due to the size of the measurement vector y, treating the problem recursively can yield a reasonable tradeoff between computational complexity and performance. Detailed surveys over nonlinear recursive estimation can be found in Sorenson [137], and Kulhavý [106].

In recursive estimation problems the parameters are usually referred to as *states* of the system and they may change with time. Let x_t denote the state at time t, where we always let the time index $t \in \mathbb{N}$ independently of the actual time evolved. This will allow for unequally spaced time steps within the same framework while keeping the notation streamlined. Uniformly spaced time intervals will however be assumed unless stated differently. The notation T will be used for the fixed sampling interval.

If a correct modeling of the problem has been performed, the states x_t are chosen such that they describe everything about the system at the present time. Hence, conditioned on the present state no additional information about the future states should be available in past observations. This Markovian property of the system states is fundamental to recursive estimation. The following proposition defines the type of recursive estimation problems that we study in this thesis.

Proposition 3.2
A recursive estimation problem is an estimation problem where the states evolve in time according to a Markov process with an initial state $x_0 \sim p(x_0)$ and transition kernel,

$$p(x_{t+1} \,|\, x_t) \qquad t = 0, 1, \ldots$$

This transition kernel may explicitly depend on the time index. The measurement observed at the time instant t is conditionally independent of the previously observed measurements given the current state value. The likelihood

$$p(y_t \,|\, x_t) \qquad t = 0, 1, \ldots \tag{3.15}$$

may explicitly depend on the time index t.

A recursive estimation problem is uniquely specified when the prior $p(x_0)$, the transition kernel $p(x_{t+1} \,|\, x_t)$ and the likelihood $p(y_t \,|\, x_t)$ are given. Note that this coincides with Proposition 3.1 since these entities define the joint density of all measurements and all states from time zero to time t. An example of a Bayesian formulation of a recursive estimation problem follows, its Bayesian solution is illustrated in the subsequent section.

Example 3.2
In the bearings-only tracking problem the objective is to track the movements of an object using measurements of the relative angle to the object. Consider, e.g., a case when the position of a ship is tracked using passive sonar measurements. Let

$$x_t = \begin{bmatrix} x_t^{(1)} \\ x_t^{(2)} \end{bmatrix}$$

be the north and east coordinates of the ship, and y_t the bearings measurement from the sonar. A simple model of the inference problem with a random ship movement is given by

$$\begin{aligned} x_{t+1} &= x_t + w_t \\ y_t &= \tan^{-1}(x_t^{(2)}/x_t^{(1)}) + e_t \end{aligned}$$

where the noises w_t and e_t are independent, white processes with density functions

$$\begin{aligned} p(w_t) &= \mathrm{N}\left(w_t; \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 200^2 & 0 \\ 0 & 200^2 \end{bmatrix}\right) \\ p(e_t) &= \mathrm{N}(e_t; 0, 0.1^2) \end{aligned}$$

Furthermore, let the prior knowledge about the ship position be

$$p(x_0) = \mathrm{N}\left(x_0; \begin{bmatrix} -2000 \\ 3000 \end{bmatrix}, \begin{bmatrix} 500^2 & 0 \\ 0 & 500^2 \end{bmatrix}\right),$$

and let it be independent of the noises acting on the system. It is a recursive estimation problem to determine the ship position x_t, given all collected data from initial time to present. A model specification in accordance with Proposition 3.2 is given by

$$\begin{aligned} p(x_{t+1} \,|\, x_t) &= \mathrm{N}\left(x_{t+1}; x_t, \begin{bmatrix} 200^2 & 0 \\ 0 & 200^2 \end{bmatrix}\right) \\ p(y_t \,|\, x_t) &= \mathrm{N}\left(y_t; \tan^{-1}(x_t^{(2)}/x_t^{(1)}), 0.1^2\right) \end{aligned}$$

and $p(x_0)$ given above.

3.5.1 Conceptual Solution

All recursive estimation problems that can be expressed in the form of Proposition 3.2 have a common conceptual solution consisting of a recursive propagation of the conditional density. The stacked vector of the complete measurement history at time t has length $(t+1)p$ and is denoted by

$$\mathbb{Y}_t = \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_t \end{bmatrix}, \qquad \text{and generally} \qquad \mathbb{Y}_{s:t} = \begin{bmatrix} y_s \\ y_{s+1} \\ \vdots \\ y_t \end{bmatrix}$$

for any $s < t$. This notation will be adopted generally for stacked vectors of random processes. Applying Bayes' rule (3.1) to the last element of the measurement vector $\mathbb{Y}_t$ yields

$$\begin{aligned} p(x_t \,|\, \mathbb{Y}_t) &= \frac{p(y_t \,|\, x_t, \mathbb{Y}_{t-1}) p(x_t \,|\, \mathbb{Y}_{t-1})}{p(y_t \,|\, \mathbb{Y}_{t-1})} \\ &= \frac{p(y_t \,|\, x_t) p(x_t \,|\, \mathbb{Y}_{t-1})}{p(y_t \,|\, \mathbb{Y}_{t-1})} \end{aligned} \tag{3.16}$$

since, by (3.15), we assume that the observation y_t is conditionally independent of previous measurements given the state x_t. The expression (3.16) is referred to as the *measurement update* in the Bayesian recursion. As in (3.1) the denominator can be expressed through the law of total probability, i.e., by marginalization.

The effect of a time step is obtained by observing that

$$\begin{aligned} p(x_{t+1}, x_t \,|\, \mathbb{Y}_t) &= p(x_{t+1} \,|\, x_t, \mathbb{Y}_t)\, p(x_t \,|\, \mathbb{Y}_t) \\ &= p(x_{t+1} \,|\, x_t)\, p(x_t \,|\, \mathbb{Y}_t) \end{aligned}$$

which follows from the assumption that the process $\{x_t\}$ is Markovian, and that x_{t+1} is independent of $\mathbb{Y}_t$ when x_t is given. Integrating both sides with respect to x_t yields the *time update* equation

$$p(x_{t+1} \,|\, \mathbb{Y}_t) = \int_{\mathbb{R}^n} p(x_{t+1} \,|\, x_t) p(x_t \,|\, \mathbb{Y}_t)\, dx_t. \tag{3.17}$$

This relation is also referred to as the *Chapman-Kolmogorov* equation [94]. After (3.17) has been evaluated, the time index can be increased and the effect of a new measurement incorporated as in (3.16). To summarize, a recursive propagation of the posterior filter density of the states given the measurements is obtained.

Theorem 3.3
A recursive estimation problem of the form in Proposition 3.2 has a recursive Bayesian solution

$$\begin{aligned} p(x_t \,|\, \mathbb{Y}_t) &= \frac{p(y_t \,|\, x_t) p(x_t \,|\, \mathbb{Y}_{t-1})}{p(y_t \,|\, \mathbb{Y}_{t-1})} \\ p(x_{t+1} \,|\, \mathbb{Y}_t) &= \int_{\mathbb{R}^n} p(x_{t+1} \,|\, x_t) p(x_t \,|\, \mathbb{Y}_t)\, dx_t \end{aligned} \qquad t = 0, 1, \ldots$$

where

$$p(y_t \,|\, \mathbb{Y}_{t-1}) = \int_{\mathbb{R}^n} p(y_t \,|\, x_t) p(x_t \,|\, \mathbb{Y}_{t-1})\, dx_t$$

and the recursion is initiated by $p(x_0 \,|\, \mathbb{Y}_{-1}) = p(x_0)$.

Commonly used point estimates derived from the posterior density are

$$x_{t|t}^{\mathrm{MS}} = \int_{\mathbb{R}^n} x_t p(x_t \mid \mathbb{Y}_t)\, dx_t$$
$$x_{t|t}^{\mathrm{MAP}} = \arg\max_{x_t} p(x_t \mid \mathbb{Y}_t),$$

compare to (3.6) and (3.7). Their respective estimation error covariance is

$$P_{t|t} = \int_{\mathbb{R}^n} (x - \hat{x})(x - \hat{x})^T p(x_t \mid \mathbb{Y}_t)\, dx_t.$$

One often distinguishes between the problems of *filtering* and *prediction* in recursive estimation. The solution to the filtering problem is given in Theorem 3.3 while the solution to the prediction problem is given by performing extra time updates (3.17).

As noted in the title of this subsection, the recursive propagation of the posterior density in Theorem 3.3 is only a conceptual solution. In general, the multidimensional integrals in Theorem 3.3 and in the calculation of estimates and the error covariances above have no explicit analytical solutions. Only when all these integrals have analytical solutions for any admissible measurement sequence $\mathbb{Y}_t$ and for all times t can the recursive Bayesian solution be solved analytically. The posterior density is then said to be parameterized by a finite dimensional *sufficient statistic*. For general models with nonlinear and non-Gaussian elements, this happens only very rarely. Hence, in a practical application of Theorem 3.3 one is forced to work with some approximate description of the posterior filter density. The format of this approximation should be well-suited for propagation through Theorem 3.3 and for calculation of the chosen estimate and its error covariance. The choice of numerical approximation will be handled in detail in Chapter 5.

We conclude this subsection with an illustration of the Bayesian solution to a common target tracking problem.

Example 3.3 (Continued from Example 3.2)
Inserting the model of the bearings-only tracking problem from Example 3.2 into the Bayesian solution in Theorem 3.3 yields

$$p(x_t \mid \mathbb{Y}_t) \propto \mathrm{N}\Big(y_t; \tan^{-1}(x_t^{(2)}/x_t^{(1)}), 0.1^2\Big)\, p(x_t \mid \mathbb{Y}_{t-1})$$
$$p(x_{t+1} \mid \mathbb{Y}_t) = \int_{\mathbb{R}^2} \mathrm{N}\left(x_{t+1}; x_t, \begin{bmatrix} 200^2 & 0 \\ 0 & 200^2 \end{bmatrix}\right) p(x_t \mid \mathbb{Y}_t)\, dx_t$$

initiated by

$$p(x_0 \mid \mathbb{Y}_{-1}) = \mathrm{N}\left(x_0; \begin{bmatrix} -2000 \\ 3000 \end{bmatrix}, \begin{bmatrix} 500^2 & 0 \\ 0 & 500^2 \end{bmatrix}\right).$$

Figure 3.1 shows a simulation with three snapshots of a numerical approximation to the conditional density. The figure shows numerical approximations to the prior $p(x_0 \mid \mathbb{Y}_{-1})$, and the posterior $p(x_t \mid \mathbb{Y}_t)$ at two stages of the recursion. Naturally, the posterior density is narrow seen from the platform but widespread along the line-of-sight between ship and platform.

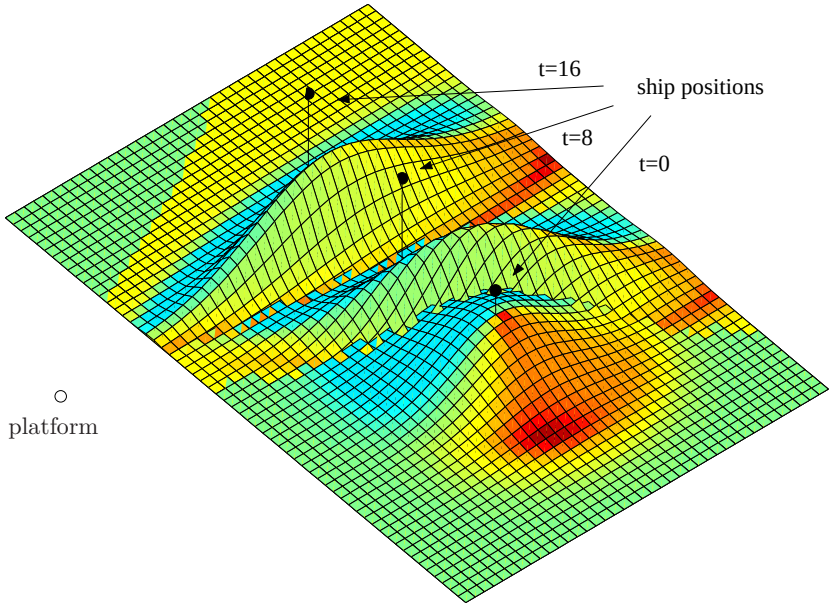


Figure 3.1: The Bayesian solution to bearings only target tracking. Three snapshots of the filtering density are shown. The prior, the filter density at time $t = 8$, and the filter density at time $t = 16$ are depicted. The ship moves from right to left and its true position at the considered time instants is indicated by solid circles.

3.5.2 Linear Recursive Estimation

The celebrated Kalman filter [98] is the Bayesian solution to recursive estimation for a specific class of models. When the measurement relation and the state transition equation both are linear and the noises acting on the system are Gaussian, an analytical solution to the recursive propagation of Theorem 3.3 exists.

Theorem 3.4 (The Kalman Filter [2, 96])
Consider the linear state space model,

$$\begin{aligned} x_{t+1} &= F_t x_t + G_t w_t \\ y_t &= H_t x_t + e_t \end{aligned} \qquad t = 0, 1, \ldots \tag{3.18}$$

where both the noises and the initial state are Gaussian distributed, the noises are white, and

$$\mathsf{E}\left(\begin{bmatrix} \boldsymbol{w}_t \\ \boldsymbol{e}_t \\ \boldsymbol{x}_0 \end{bmatrix} \begin{bmatrix} \boldsymbol{w}_t^T & \boldsymbol{e}_t^T & \boldsymbol{x}_0^T & 1 \end{bmatrix} \right) = \begin{bmatrix} Q_t & S_t & 0 & 0 \\ S_t^T & R_t & 0 & 0 \\ 0 & 0 & P_0 & x_0 \end{bmatrix}.$$

The conditional density for the one-step ahead prediction problem, $p(x_t \mid \mathbb{Y}_{t-1})$, is Gaussian. Initiated by $\hat{x}_0 = x_0$, the mean obeys the recursion

$$\hat{x}_{t+1} = F_t\hat{x}_t + K_t(y_t - H_t\hat{x}_t),$$

where $K_t = (F_tP_tH_t^T + G_tS_t)(R_t + H_tP_tH_t^T)^{-1}$ and the posterior covariance is

$$P_{t+1} = F_tP_tF_t^T - K_t(R_t + H_tP_tH_t^T)K_t^T + G_tQ_tG_t^T$$

initiated by P_0.

Proof A proof of the statement above along with comprehensive presentations regarding all aspects of Kalman filtering can be found in, e.g., [2, 96]. □

For the linear and Gaussian model class described above, Theorem 3.1 yields that the posterior distribution is Gaussian. Hence, a sufficient statistic consists of the mean vector and the covariance matrix of this Gaussian conditional density. It is these parameters that are given by the analytical solution to the Bayesian recursive estimation problem in the theorem above. Due to the properties of the Gaussian density, the recursively updated parameters also coincide with the mean-square estimate and its error covariance for the linear Gaussian model class. The Kalman filter has several interpretations apart from being a Bayesian solution to a certain class of models, see [2, 96] for a thorough presentation of linear estimation theory.

3.5.3 Nonlinear Recursive Estimation

Most real-world problems have elements of nonlinearity, non-Gaussianity and non-stationarity. For this large class of practical problems it is not possible to derive exact closed form solutions for the estimators and estimation error covariances. The most common approach to circumvent this problem has been to resort to model simplifications and approximations which inevitably will lead to performance degradations. In critical situations where one cannot compromise the accuracy of the result, the only way to solve these inference problems is by numerical approximation techniques. A few notable exceptions of non-standard models with exact finite dimensional solutions exist, see, e.g., [65, 105]. For practical nonlinear estimation, approximative solutions must in general be considered. These approximations can be categorized into two groups, either the underlying model of the problem is simplified so that an analytical solution is obtained, or an approximative description of the posterior distribution itself is determined.

Local model approximations

The approach to locally approximate the nonlinear model is the most frequently used practical technique for nonlinear recursive estimation. The nonlinear model equations are expanded into a Taylor series around a point estimate of the states. A model of the form (3.18) is obtained by inserting the estimate obtained from

the last iteration of the algorithm, and approximating the distributions of noises acting on this linearized model with Gaussian distributions. The system matrices of this linear time dependent model will explicitly depend on the estimate that the linearization is performed around. Disregarding this dependence, the Kalman filter can be applied directly to this linearized time dependent model. This scheme is usually labeled the *Extended Kalman Filter* (EKF) [2, 94]. Several applications of EKF techniques to nonlinear estimation problems can be found in Sorenson [136].

The EKF approach yields a nonlinear estimation filter with very modest computational requirements that is frequently applied in practical recursive nonlinear estimation such as target tracking, navigation, and robotics. However, there are some well known disadvantages with this linearization technique. It often happens that the estimation error P_t, calculated by the algorithm, becomes inconsistent with the true estimation error. Usually, the filter underestimates this error resulting in a reduction of the filter gain and an insensitivity to new measurements. This effect is labeled the *divergence* problem of the EKF. The situation usually arises as a result of linearization errors when the local approximation ceases to hold. Apart from manually keeping the filter gain above a certain level, several generalizations that deal with the divergence problem have been proposed. These include methods for higher order approximation schemes and the iterated version of the EKF. Jazwinski [94] gives a thorough treatment of these topics.

Another sort of model approximation is obtained by letting the state vector be limited to a finite number of values. In this case the conditional density recursion can be solved by approximately determining the conditional probability for each candidate state value using numerical integration methods. This type of model with discrete valued states is often referred to as a Hidden Markov Model (HMM). A comprehensive presentation of HMMs, containing results regarding both estimation and control, can be found in Elliott et al. [63], and a tutorial over HMMs is given by Rabiner [125].

Global approximation of the posterior density

The local linearization technique of the EKF fails to hold when the nonlinearity of the model is substantial, the initial estimate is bad or the noises acting on the model are far from Gaussian distributed. A global approximation is obtained by keeping the original nonlinear model but instead solving the Bayesian recursion of Theorem 3.3 approximately. There are two common approaches to global approximation of the Bayesian solution. Either the posterior density is parameterized in a finite number of parameters that are inserted into the Bayesian recursion and approximately updated with each iteration, or the integrals in the recursion are directly attacked by numerical integration methods. Both these approaches result in similar globally approximate Bayesian filters. With no linearization errors introduced, these methods promise a solution closer to the optimal one. Contrary to the EKF technique, these filters may, e.g., handle multi-modal conditional densities without any further modifications. However, this is usually obtained at the price of a substantially higher computational burden.

Numerical integration methods based on grid approximations of the posterior density will be presented in more detail in Chapter 5. In many cases, the resulting algorithms of numerical integration will be rather similar to the approach of approximating the conditional density. Generally, practical on-line estimation is a function approximation problem under the constraints that an efficient and reliable update of the function expansion through the Bayesian recursion is possible. The measurement and time updates consist of multiplications and convolutions between probability density functions. From the analytical solution of the linear Gaussian model, it is known that multiplying and convolving Gaussian distributions will generate new Gaussian distributions. A reasonable approximative expansion of the posterior is therefore obtained by a weighted sum of Gaussian distributions,

$$p(x_t \,|\, \mathbb{Y}_{t-1}) \approx \sum_{i=1}^{N} \gamma_i \, \mathrm{N}(x_t; \mu_i, \Sigma_i). \tag{3.19}$$

This is utilized in the *Gaussian sum* estimation techniques introduced by Alspach and Sorenson [1], Sorenson and Alspach [138]. Through the measurement update multiplication, the number of terms in the approximating sum (3.19) will in general increase at each iteration. In [1, 138] it is therefore suggested that terms with small weights should be neglected and terms that are close should be combined into a single term at each iteration of the algorithm.

Another expansion that lends itself to the nonlinear estimation problem is the *point-mass approach*, originally suggested by Bucy and Senne [34]. In the point-mass approach the posterior is approximated by a set of point values on a moving grid

$$p(x_t \,|\, \mathbb{Y}_t) \approx \sum_{i=1}^{N} p_i \delta(x_t - x_t^i).$$

With this approach, the integrals will be replaced by summations that can be suitably truncated over finite intervals. The point-mass approximation has been generalized to piecewise constant approximations by Kramer [102], Kramer and Sorenson [104]. This approach to nonlinear filtering was developed during the early 1970's. At this time, the main concern about the practical applicability of these techniques was the problem of storing the comparatively large number of grid points. Due to this constraining limitation, research focused on more advanced interpolation techniques that could produce the same results with fewer grid points. The spline filters of de Figueiredo and Jan [47] and Youssef [155] were developed with this in mind. The more advanced interpolation schemes in general suffer from a more complicated description of the grid nodes. The propagation of the grid introduces extra overhead computations that, to a large extent, compensate for the reduction of the number of interpolation functions. A thorough background to the development of global approximation methods in nonlinear estimation with an emphasis on point-mass approximations is presented in [137]. Lately, the point-mass approaches have had a minor revival due to the rapid increase in desktop

computer memory size and floating point computational performance. Simandl and Královec [133] give some general guidelines towards choosing the grid resolution and support.

A somewhat different approach to approximation of the posterior distribution is taken by Kulhavý [106]. From the empirical density of the observations, a parametric posterior distribution from an exponential family is determined using convex optimization methods. This optimization can be posed such that each new measurement is added recursively and an on-line implementation is thus possible. Kulhavý only considers inference of nonrandom parameters but introduces the possibility of tracking time-varying parameters using forgetting factor techniques. Applications to several recursive estimation problems and to adaptive control problems are given in [106].

3.5.4 Continuous Time Filtering

Above, and throughout this thesis, we study the recursive estimation problem in discrete time only. Hence, the Markovian transitions of the states is assumed to evolve in discrete steps and the measurements are obtained at discrete time instants. One can argue that the class of purely discrete time models, although possibly nonlinear, is too restrictive for practical application. Often, real world phenomena evolve in continuous time while measurements are collected at discrete time instants. Consider, e.g., the target tracking problem where the position and velocity of an aircraft target change continuously in time while the radar measurements are obtained periodically, when the target is illuminated by the rotating radar antenna.

With continuous time state evolution, the Chapman-Kolmogorov equation (3.17) will turn into a partial differential equation, often referred to as the *Kolmogorov forward equation* or the *Fokker-Plank* equation [94]. This nonlinear partial differential equation has been studied extensively in the field of optimal filtering. Apart from the obvious linear Gaussian case, analytical solutions to the Fokker-Plank equation have only been found in rare cases [13, 43]. For general nonlinear and non-Gaussian models, EKF linearization techniques can be applied similarly to the discrete time case. The counterpart to the numerical integration approach of discrete time filtering is numerical solution of the nonlinear Fokker-Plank partial differential equation. A general adaptive grid method for this problem has been developed by Cai et al. [36]. The techniques are similar to the nonlinear integration methods for discrete time filtering and in general very time consuming and application specific.

From almost any continuous time evolution, a purely discrete time model can be obtained by performing several small time update steps between each measurement update. For this reason, we will solely consider discrete time estimation problems and for simplicity that only one time update is needed between each measurement update. A thorough treatment of discrete and continuous time nonlinear filtering and estimation as well as hybrid problems with, e.g., continuous time state evolution and discrete time measurements, is given by Jazwinski [94].

3.6 Summary

Many practical problems in signal processing can be described accurately by an underlying statistical model of the problem. In this context, the Bayesian approach to statistical inference gives a natural framework where information given as prior knowledge as well as information in observations are considered. Both parameters and observations are regarded as random variables and inference is performed by conditioning through Bayes' rule.

The posterior density $p(x \mid y)$ summarizes the information gained after observing the experimental outcome y. The mean of this function $\hat{x}_{\mathrm{MS}}$ is an estimate that minimizes the Bayesian risk for a large class of cost functions. The error covariance of $\hat{x}_{\mathrm{MS}}$ is equal to the covariance of the random variable $x \mid y$ and is useful when comparison or combination with other estimates should be performed.

The expressions for the normalizing factor of the posterior $p(x \mid y)$, the estimate and estimation error covariance, all involve multidimensional integrals or optimizations. Unfortunately, it is only in certain special cases these integrals and optimizations have analytical solutions. In a recursive setting the Bayesian solution consists of sequentially updating the posterior density with each new observation and with time. This posterior can only rarely be parameterized using a finite dimensional sufficient statistic which may be sequentially updated analytically. Practical recursive Bayesian estimation therefore demands numerical solutions to high dimensional complex integrations or optimizations with respect to the parameters. In addition to solving these integration problems, the posterior density itself must be approximately described in a way that lends itself to approximate propagation through the conceptual Bayesian solution.

Appendix

3.A Foundational Concepts of Probability Theory

This appendix summarizes the basic concepts of probability theory, from Kolmogorov's definition of probability to conditional probability density functions and Bayes' law. It also serves to settle the notation used in the thesis. The appendix relies on several texts, the main reference is [120]. The presentation herein is by no means complete or comprehensive. Thorough treatments of the topics presented in this appendix with complete proofs, enlightening discussions and several additional concepts can be found in either of [40, 109, 120].

3.A.1 Probability Theory

The basic concepts of probability theory are the event space Ω, which consists of all possible outcomes of an experiment, and events which are subsets of Ω. The event space Ω is also referred to as the certain event, while the empty set, $\emptyset$, is the impossible event. Events that are singletons are sometimes denoted ω.

Definition 3.A.1 (Axiomatic definition of probability)
To each event $\mathcal{A}$ we assign the number $\Pr(\mathcal{A})$ which we call the probability of the event $\mathcal{A}$. The probability is chosen to satisfy the following four axioms:

1. $\Pr(\mathcal{A}) \geq 0$
2. $\Pr(\Omega) = 1$
3. *If* $\mathcal{A} \cap \mathcal{B} = \emptyset$, *then* $\Pr(\mathcal{A} \cup \mathcal{B}) = \Pr(\mathcal{A}) + \Pr(\mathcal{B})$.
4. *If* $\mathcal{A}_i \cap \mathcal{A}_j = \emptyset$ *for* $i \neq j$, *then* $\Pr(\cup_{i=1}^{\infty} \mathcal{A}_i) = \sum_{i=1}^{\infty} \Pr(\mathcal{A}_i)$

All statistical inference is based on the concept of conditional probabilities. A conditional probability describes what effect knowing a certain event has on the probabilities of other, related, events.

Definition 3.A.2
The conditional probability of an event $\mathcal{A}$ given an event $\mathcal{M}$ of nonzero probability $\Pr(\mathcal{M}) > 0$ is defined as the ratio

$$\Pr(\mathcal{A} \,|\, \mathcal{M}) = \frac{\Pr(\mathcal{A} \cap \mathcal{M})}{\Pr(\mathcal{M})}$$

It is readily possible to verify that a conditional probability is a probability in accordance with the axioms of Definition 3.A.1. From this we can deduce that the probability of every event can be written as a sum of conditional probabilities.

Theorem 3.A.5 (Total probability theorem)
Let $\{\mathcal{A}_1, \ldots, \mathcal{A}_n\}$ be a partition of Ω and let $\mathcal{B}$ be an arbitrary event, then

$$\Pr(\mathcal{B}) = \Pr(\mathcal{B} \,|\, \mathcal{A}_1) \Pr(\mathcal{A}_1) + \cdots + \Pr(\mathcal{B} \,|\, \mathcal{A}_n) \Pr(\mathcal{A}_n)$$

Combining Definition 3.A.2 and Theorem 3.A.5 yields the following famous formula for conditional probability, originally due to Bayes [9] in 1763.

Theorem 3.A.6 (Bayes' theorem)
Let $\{\mathcal{A}_1, \ldots, \mathcal{A}_n\}$ be a partition of Ω and let $\mathcal{B}$ and $\mathcal{M}$ be two events, then

$$\Pr(\mathcal{M} \,|\, \mathcal{B}) = \frac{\Pr(\mathcal{B} \,|\, \mathcal{M}) \Pr(\mathcal{M})}{\Pr(\mathcal{B} \,|\, \mathcal{A}_1) \Pr(\mathcal{A}_1) + \cdots + \Pr(\mathcal{B} \,|\, \mathcal{A}_n) \Pr(\mathcal{A}_n)}$$

3.A.2 Random Variables

A random variable is a real valued function with domain Ω, the set of all possible experimental outcomes. For every outcome ω we assign a real number $\boldsymbol{x}(\omega)$. We will often omit the explicit dependency on the event ω and write the random variable as a bold character $\boldsymbol{x}$. Due to the frequent use of stochastic variables in this thesis we will even alleviate the bold character notation when no risk for ambiguity exists. The set $\{\boldsymbol{x} \leq x\}$ is a set in the event space, a subset of Ω consisting of all outcomes ω such that $\boldsymbol{x}(\omega) \leq x$. The *distribution function* of $\boldsymbol{x}$ is the function

$$P_x(x) = \Pr(\boldsymbol{x} \leq x),$$

and the *density function* is the derivative of the distribution function

$$p_x(x) = \frac{dP(x)}{dx}.$$

When no risk of ambiguity exists, we will omit the subindex in the notation $P_x(x)$ and $p_x(x)$, and simply write $P(x)$ and $p(x)$.

Definition 3.A.2 of conditional probability is straightforwardly extended to distribution functions.

Definition 3.A.3 (Conditional distribution)
The conditional distribution *of a random variable* $\boldsymbol{x}$ *given an event* $\mathcal{M}$ *of nonzero probability* $\Pr(\mathcal{M}) > 0$ *is*

$$P(x \,|\, \mathcal{M}) = \Pr(\boldsymbol{x} \leq x \,|\, \mathcal{M}) = \frac{\Pr(\boldsymbol{x} \leq x, \mathcal{M})}{\Pr(\mathcal{M})}.$$

The event $\{\boldsymbol{x} \leq x, \mathcal{M}\}$ *is the event consisting of all outcomes* ω *such that* $\boldsymbol{x}(\omega) \leq x$ *and* $\omega \in \mathcal{M}$.

The *conditional density* is the derivative of $P(x \,|\, \mathcal{M})$,

$$p(x \,|\, \mathcal{M}) = \frac{dP(x \,|\, \mathcal{M})}{dx} = \lim_{\Delta x \to 0} \frac{P(x \leq \boldsymbol{x} \leq x + \Delta x \,|\, \mathcal{M})}{\Delta x}$$

The conditional probability of the event $\mathcal{A}$ given an event of zero probability, like $\{\boldsymbol{x} = x\}$, is defined through the limit

$$\Pr(\mathcal{A} \,|\, \boldsymbol{x} = x) \triangleq \lim_{\Delta x \to 0} \Pr(\mathcal{A} \,|\, x \leq \boldsymbol{x} \leq x + \Delta x). \tag{3.A.20}$$

The total probability theorem, Theorem 3.A.5, for conditional events of zero probability can then be expressed as an infinite sum over the limits in (3.A.20). Hence, the probability of any event $\mathcal{A}$ can be written

$$\Pr(\mathcal{A}) = \int_{-\infty}^{\infty} \Pr(\mathcal{A} \,|\, \boldsymbol{x} = x) p(x)\, dx.$$

The conditional density of $\boldsymbol{x}$ given the event $\mathcal{A}$ is

$$p(x \,|\, \mathcal{A}) = \frac{\Pr(\mathcal{A} \,|\, \boldsymbol{x} = x)\, p(x)}{\Pr(\mathcal{A})}. \tag{3.A.21}$$

So far we have considered events and real scalar random variables. The extension to vector valued random variables follows conceptually by looking at pairs of scalar variables. The distribution and density functions extend naturally as

$$\begin{aligned} P(x, y) &= \Pr(\boldsymbol{x} \leq x, \boldsymbol{y} \leq y) \\ p(x, y) &= \frac{\partial^2 P(x, y)}{\partial x\, \partial y}. \end{aligned}$$

These functions are occasionally given the prefix joint distribution and density function, respectively. The definition of conditional distribution and density applies to pairs of random variables as well

$$\begin{aligned} P(x, y \,|\, \mathcal{M}) &= \frac{\Pr(\boldsymbol{x} \leq x, \boldsymbol{y} \leq y, \mathcal{M})}{\Pr(\mathcal{M})} \\ p(x, y \,|\, \mathcal{M}) &= \frac{\partial^2 P(x, y \,|\, \mathcal{M})}{\partial x\, \partial y} \end{aligned} \tag{3.A.22}$$

As in Definition 3.A.3, it is assumed that $\Pr(\mathcal{M}) > 0$. The conditional density of a random variable $\boldsymbol{y}$, assuming the value of some other random variable $\boldsymbol{x}$ is known, is frequently used in statistical inference. This density cannot be derived directly from (3.A.22) due to the fact that the event $\{\boldsymbol{x} = x\}$ generally has zero probability. Instead, this density is defined through the limit

$$p(y \,|\, \boldsymbol{x} = x) = \lim_{\Delta x \to 0} p(y \,|\, x < \boldsymbol{x} \leq x + \Delta x) = \frac{p(x, y)}{p(y)}. \tag{3.A.23}$$

When no risk of ambiguity exists, the shorter notation $p(y \,|\, x)$ will be used in liu of the somewhat cumbersome notation $p(y \,|\, \boldsymbol{x} = x)$. Combining (3.A.23) with (3.A.21) yields the most common version of Bayes' theorem.

Theorem 3.A.7 (Bayes' theorem and total probability)
The conditional density can be written

$$p(x \,|\, y) = \frac{p(y \,|\, x)\, p(x)}{p(y)} \tag{3.A.24}$$

where the normalizing constant in the denominator can be expressed using the law of total probability

$$p(y) = \int_{\mathbb{R}^n} p(y \,|\, x) p(x)\, dx.$$

The function $p(y \,|\, x)$ is labeled the *likelihood* while the density function of the parameters $p(x)$ is called the *prior*. Using the total probability theorem it is worth noting that Bayes' law (3.A.24) can be expressed solely using the joint density of $\boldsymbol{x}$ and $\boldsymbol{y}$,

$$p(x \,|\, y) = \frac{p(y, x)}{\int_{\mathbb{R}^n} p(y, x)\, dx}.$$

Since the conditional density is the fundamental solution to the inference problem, it follows that the joint density is the only description needed for doing statistical inference.

4

Cramér-Rao Bounds

The posterior filter density of most nonlinear recursive estimation problems cannot be described analytically by a finite number of parameters. Several examples of suboptimal algorithms for practical nonlinear recursive estimation have therefore appeared in the literature. Generally, these procedures approximate either the estimation model or the description of the posterior distribution. These inevitable approximations may seriously degrade the estimation performance compared to the results obtained if the posterior filter density was known exactly. It is of great practical interest to quantify this performance degradation, and measure the effect of the introduced approximations. A benchmark simulation evaluation against the optimal solution is not an option since it would require infinite computing power and unlimited memory resources to run the optimal algorithm. However, even if the optimal solution is intractable, the performance of the optimal algorithm may very often be computed with limited resources. The estimation error obtained with an optimal algorithm depends only on fundamental properties of the model, e.g., signal to noise levels and prior assumptions on the sought parameters. Characteristics of the optimal estimation error define lower limits on the performance that can be achieved using any approximative implementation. The characteristics of the suboptimal estimation error are revealed in simulation studies using the implemented procedure. The discrepancy from the lower bound gives an indication of the effect of the approximations introduced in the implemented algorithm. A relative comparison between the suboptimal and the optimal algorithms is therefore possible

even if it is intractable to implement the optimal solution. A lower bound on some property of the estimation error is convenient when evaluating implemented suboptimal procedures, but can also be used to quantify the fundamental performance level that can be reached for the currently studied estimation problem.

In this chapter we focus on the fundamental Cramér-Rao bound which sets a lower limit on the mean square estimation error. The bound states that the difference between the expected mean square error correlation matrix and a matrix determined by the model of the estimation problem, should always be positive semidefinite. The parametric and the Bayesian viewpoints on estimation give two different kinds of Cramér-Rao bounds. These bounds, known as parametric and posterior Cramér-Rao bounds respectively, are reviewed in Section 4.1. In Section 4.2, we review some results on Cramér-Rao bounds for recursive estimation, and present the class of nonlinear state space estimation models that is considered in this work. The subsequent section presents compact expressions for the parametric Cramér-Rao bound for general filtering and prediction. The smoothing case is also handled, in a somewhat rudimentary fashion. The recursive formulas for the Cramér-Rao bound hold true for general nonlinear and non-Gaussian models and require limited computational efforts and memory resources. Similar expressions are presented using the posterior bound in Section 4.4. Without restrictions that guarantee non-singularities in the state evolution process, the computations and storage requirements for the posterior Cramér-Rao bound for recursive estimation will increase linearly with time. The class of estimation problems for which the posterior Cramér-Rao bound can be computed with limited effort still includes general nonlinear models with possibly non-Gaussian noises. In Section 4.5, we present some suggestions to the practical application of the Cramér-Rao bounds. The utilization of both the parametric and the posterior Cramér-Rao bound is also exemplified by the terrain navigation application in Chapter 7. A summary of the results and some conclusive remarks are finally given at the end of this chapter.

4.1 Review over Cramér-Rao Bounds

In this section, we review the Cramér-Rao bound for the generic estimation problem discussed in Section 3.2. The sought parameters are assembled in an n-dimensional vector x and the observations are given as a p-dimensional vector y. For random parameters, the estimation relies on the Bayesian paradigm and the problem is uniquely specified by the joint density $p(x, y) = p(y \mid x)p(x)$. For nonrandom parameters, it is the likelihood $p(y \mid x)$ that alone defines the estimation problem.

For sake of completeness, the bounds reviewed in this section are presented with derivations from first principles. The technical details of the proofs are summarized in Appendix 4.B. The fact that we deliver complete derivations does not mean that we claim originality of all the results presented. The main results of this review can be found in any textbook on estimation theory. For a thorough treatment of the results in this section we refer to the excellent, and now classical, monograph of Van Trees [151].

Commonly, the Cramér-Rao bound for multiple parameter estimation is derived as an extension from the scalar case. Contrary to, e.g., Van Trees [151], we choose to present the general bounds for vector valued estimation directly. In addition, we introduce a slight generalization of the classical bounds. This modification is utilized in the application of the bounds to the recursive estimation problem. It allows us to derive expressions for the Cramér-Rao bound to a general class of state space estimation problems.

4.1.1 Estimator Performance

Consider the generic statistical inference problem of estimating an n-dimensional parameter x based on a p-dimensional observation vector y. The fundamental Cramér-Rao bound sets a lower limit on the mean square estimation error of any estimator $\hat{x}(y)$. The bound applies to all estimators under certain conditions on the estimator bias, which is defined as the expected systematic error it produces averaged over all measurements.

Definition 4.1
The bias *of an estimator $\hat{x}(y)$ is the expected error,*

$$B(x) \triangleq \int_{\mathbb{R}^p} (\hat{x}(y) - x) p(y \,|\, x)\, dy.$$

In parametric estimation, the bias vector is a function of the unknown but fixed true parameter $x = x^\star$. In the Bayesian framework, it is a random vector defined by the function $B(\cdot)$, and the prior distribution $p(x)$. Depending on the properties of the bias vector, the estimators are grouped into three different classes.

Unbiased When the bias $B(x) = 0$ for all x, the estimator is *unbiased*. On the average, the estimate equals the sought parameter.

Known bias When the bias is nonzero but independent of the parameter $B(x) = B$, the estimator is said to have a *known bias*. An unbiased estimator is straightforwardly obtained by subtracting the known bias from the estimate.

Unknown bias In the general case, $B(x)$ is a function of the unknown parameter x and the estimator is said to have an *unknown bias*. Since the bias is a function of the unknown parameter, it cannot be subtracted from the estimate.

An unbiased estimator for the nonrandom parameter x has an estimation error covariance given by

$$C = \mathsf{Cov}(\hat{x}(\boldsymbol{y}) - x) = \mathsf{E}\big((\hat{x}(\boldsymbol{y}) - x)(\hat{x}(\boldsymbol{y}) - x)^T\big)\,.$$

In parametric estimation the expectation is performed over the only stochastic variable, viz. y, using the parameterized density $p(y \,|\, x)$. Given any unbiased estimator defined for a parametric estimation problem, the Cramér-Rao bound for

this problem is a matrix P of the same dimension as the estimator error covariance C that satisfies

$$C - P \geq 0. \tag{4.1}$$

The inequality in (4.1) means that the left hand side is positive semidefinite. In the random parameter case, it turns out that the bias has less importance and that the estimation error correlation matrix, or mean square error matrix,

$$C = \mathsf{E}\big((\hat{x}(\boldsymbol{y}) - \boldsymbol{x})(\hat{x}(\boldsymbol{y}) - \boldsymbol{x})^T\big) = \mathsf{Cov}(\hat{x}(\boldsymbol{y}) - \boldsymbol{x}) + \mathsf{E}(B(\boldsymbol{x}))\,\mathsf{E}\big(B^T(\boldsymbol{x})\big)$$

satisfies the Cramér-Rao inequality (4.1), given only minor assumptions on the estimator bias. In the random case, the expectation is over both x and y, using the joint density $p(x, y)$.

The Cramér-Rao bound P is uniquely determined by the estimation model. In the case of fixed nonrandom parameters, P depends on the likelihood and the unknown fixed parameter. In the Bayesian random case, P depends on the joint density. Proof of (4.1) and expressions for P in both the case of parametric and of random estimation follow in the two subsequent sections. The material of these two subsections have mainly been collected from [151]. An alternative presentation only covering the parametric case is given by Scharf [130]. However, the Cramér-Rao bounds presented below are somewhat generalized versions of the ones found in [151]. We introduce the ability to distinguish between the parameter vector that we are interested in estimating, x, and the vector z that affects the estimation model. This makes it possible to handle cases when the prior density of x is singular, i.e., involves Dirac delta measures.

The sequel of this chapter involves some, rather standard, vector gradient notation and utilizes some basic block matrix manipulations. Appendix 4.A reviews some of the utilized results, and settles the basic notation.

4.1.2 Nonrandom Parameters

When the parameters are regarded as unknown but fixed, they define a parameterization of all admissible probability density functions for the observations, $p(y \,|\, x)$. The parametric Cramér-Rao bound depends on this likelihood and on the true unknown parameter, denoted by $x^\star$.

The *score function* is the gradient of the logarithm of the likelihood

$$\nabla_x \log p(y \,|\, x).$$

In maximum likelihood estimation, this function returns a vector valued score given the observation y and a candidate parameter vector x. Scores close to zero are good scores since they indicate that x is close to a local optimum of $p(y \,|\, x)$. Regarding $p(y \,|\, x)$ as a positive function of x,

$$\nabla_x \log p(y \,|\, x) = \frac{1}{p(y \,|\, x)} \nabla_x p(y \,|\, x).$$

Inserting the random vector $\boldsymbol{y}$ into the score function, each candidate parameter vector yields a stochastic score. The expected value of the stochastic score function is zero since

$$\begin{aligned}\mathsf{E}(\nabla_x \log p(\boldsymbol{y} \,|\, x)) &= \int_{\mathbb{R}^p} \nabla_x (\log p(y \,|\, x))\, p(y \,|\, x)\, dy = \int_{\mathbb{R}^p} \nabla_x p(y \,|\, x)\, dy \\ &= \nabla_x \int_{\mathbb{R}^p} p(y \,|\, x)\, dy = 0.\end{aligned}$$

This implies that the correlation and covariance matrices of the score function coincide. The Fisher Information Matrix (FIM) is defined as this covariance matrix of the random score vector.

Definition 4.2 (The Fisher information matrix)
The Fisher information matrix (FIM) is the covariance matrix of the score function

$$J(x) \triangleq \mathsf{E}\big(\nabla_x \log p(y \,|\, x) \nabla_x^T \log p(y \,|\, x)\big)$$

In Corollary 4.B.3 in Appendix 4.B an alternative and more compact expression for the FIM is given,

$$J(x) = \mathsf{E}(-\Delta_x^x \log p(y \,|\, x))\,, \tag{4.2}$$

where Δ_x^x is the second order partial derivative operator, see Appendix 4.A. The FIM is a measure of the average size of the random score vector, for a given parameter value x. We will often suppress the dependency on the parameter vector and, when no risk of ambiguity exists, solely write J for the FIM of the currently studied parametric estimation problem.

In parametric estimation, the probability density function of the observations is parameterized by the sought parameter vector. The stochastic model of the observation vector is uniquely defined given a parameter vector x. However, this does not imply that all elements of x or combinations of elements of x affect the stochastic properties of the observations. The likelihood $p(y \,|\, x)$ might be overparameterized so that some elements of x or combinations of elements of x do not affect $p(y \,|\, x)$, regarded as a probability density function for y. In such a case, the FIM (4.2) for the parameter vector x becomes singular. This will lead to problems when computing the Cramér-Rao bound. Let $z \in \mathbb{R}^r$ be an alternative parameterization of the likelihood such that $p(y \,|\, z)$ is a well defined density function for y given any $z \in \mathbb{R}^r$ and the corresponding FIM is nonsingular. Hence, we let $r \leq n$ and define the possibly noninvertible coordinate transformation $x = t(z)$. This possibility to distinguish between the quantity that is estimated, x, and the parameters that directly affect the likelihood, z, is utilized in the subsequent sections when the bound is applied to recursive nonlinear estimation. The Cramér-Rao bound for parametric estimators $\hat{x}(y)$ given a possibly overparameterized likelihood follows.

Theorem 4.1 (Parametric Cramér-Rao bound)
Assume that the observation $y \in \mathbb{R}^p$ has a well defined probability density function $p(y \mid z)$ for all $z \in \mathbb{R}^r$, and let $z^\star$ denote the parameter that yields the true distribution of y. Moreover, let $\hat{x}(y) \in \mathbb{R}^n$ be an unbiased estimator of $x = t(z)$, and let $x^\star = t(z^\star)$. The estimation error covariance of $\hat{x}(y)$ is bounded from below by

$$\mathsf{E}\big((\hat{x} - x^\star)(\hat{x} - x^\star)^T\big) \geq M J^{-1} M^T$$

where

$$J = \mathsf{E}(-\Delta_z^z \log p(y \mid z))\Big|_{z=z^\star} \qquad \text{and} \qquad M^T = \nabla_z t^T(z)\Big|_{z=z^\star}$$

are matrices that depend on the true unknown parameter vector $z^\star$.

Proof A detailed proof is given in Appendix 4.B.1. In essence, the bound follows from the positive semi-definiteness of the covariance matrix

$$\mathsf{E}\left(\begin{bmatrix} \hat{x} - x \\ \nabla_z \log p(y \mid z) \end{bmatrix} \begin{bmatrix} \hat{x} - x \\ \nabla_z \log p(y \mid z) \end{bmatrix}^T\right) = \begin{bmatrix} C & M \\ M^T & J \end{bmatrix} \geq 0.$$

The Schur complement of C in this block matrix gives the desired result. □

Inserting the identity transformation $x = t(z) = z$ yields the classical Cramér-Rao bound which says that the estimation error covariance of any unbiased estimator is bounded from below by the inverse of the FIM. The general bound given in Theorem 4.1, with the possibility to have a gradient matrix $M \neq I$, is commonly not utilized in the current literature on Cramér-Rao bounds. We believe this generalization to be of great practical importance since the formulation above allows us to evaluate the bound in cases when the FIM is singular. A suitable reformulation of a singular problem by parameterizing the likelihood with respect to fewer parameters can make the FIM invertible. Theorem 4.1 yields a bound for the original likelihood parameterization by utilizing the transformation between the different likelihood parameterizations. This aspect is crucial for the general Cramér-Rao bounds to recursive estimation presented in the latter part of this chapter.

The general bound given in Theorem 4.1 can also be used to obtain a Cramér-Rao bound for estimators with possibly nonzero bias.

Corollary 4.1 (Biased Estimators)
Consider an estimation problem defined by the likelihood $p(y \mid z)$, and the fixed unknown parameter $z^\star$. Any estimator $\hat{z}(y)$ with unknown bias $B(z)$ has a mean square error bounded from below by

$$\mathsf{E}\big((\hat{z} - z^\star)(\hat{z} - z^\star)^T\big) \geq M J^{-1} M^T + B(z^\star) B^T(z^\star)$$

where

$$J = \mathsf{E}(-\Delta_z^z \log p(y \mid z))\Big|_{z=z^\star} \qquad \text{and} \qquad M^T = I + \nabla_z B^T(z)\Big|_{z=z^\star}$$

Proof Introducing the quantity $x = z + B(z)$, the estimator $\hat{x}(y) \triangleq \hat{z}(y)$ is an unbiased estimator of x. Hence, Theorem 4.1 yields that

$$\mathsf{E}\Big((\hat{x} - x)(\hat{x} - x)^T\Big) \geq (I + \nabla_z B^T(z))^T \left\{\mathsf{E}(-\Delta_z^z \log p(y \,|\, z))\right\}^{-1} (I + \nabla_z B^T(z))$$

and thus that

$$\mathsf{E}\Big((\hat{z} - z)(\hat{z} - z)^T\Big) \geq (I + \nabla_z B^T(z))^T \left\{\mathsf{E}(-\Delta_z^z \log p(y \,|\, z))\right\}^{-1} (I + \nabla_z B^T(z)) + B(z)B^T(z),$$

after suitably inserting the true unknown parameter $z^\star$. □

The Cramér-Rao bound in Theorem 4.1 depends on the true unknown parameter vector $z^\star$, and on the model of the problem defined by $p(y \,|\, z)$ and the mapping $t(z)$. Hence, the bound can typically only be computed in simulation studies, when the true value of the sought parameter vector is known.

4.1.3 Random Parameters

For random parameters there is no true unknown parameter value. Instead, the prior assumption on the parameter distribution determines the probability of different parameter vectors. Like in the parametric case of the previous subsection, we allow for a possibly non-invertible mapping $t : \mathbb{R}^r \to \mathbb{R}^n$ between a parameter vector z and the sought parameter x. The vector z is assumed to have been chosen such that the joint density $p(z, y)$ is a well defined density, without singular Dirac delta factors. The Cramér-Rao bound presented in the theorem below is a slight generalization of the posterior Cramér-Rao bound for multiple parameter estimation by Van Trees [151].

Theorem 4.2 (Posterior Cramér-Rao bound)
Let $z \in \mathbb{R}^r$ and $y \in \mathbb{R}^p$ be two random vectors with a well defined joint density $p(z, y)$, and let $\hat{x}(y) \in \mathbb{R}^n$ be an estimate of $x = t(z)$. If the estimator bias

$$B(z) = \int_{\mathbb{R}^p} (\hat{x} - t(z))\, p(y \,|\, z)\, dy$$

satisfies

$$\lim_{z_i \to \pm\infty} B_j(z)p(z) = 0 \qquad \text{for all } i = 1, \ldots, r \text{ and } j = 1, \ldots, n,$$

then the mean square error of the estimate is bounded from below

$$\mathsf{E}\big((\hat{x} - x)(\hat{x} - x)^T\big) \geq M J^{-1} M^T$$

where

$$J = \mathsf{E}(-\Delta_z^z \log p(y, z)) \qquad \text{and} \qquad M^T = \mathsf{E}\big(\nabla_z t^T(z)\big)\,.$$

Proof The bound follows from the positive semidefinite correlation matrix

$$\mathsf{E}\left(\begin{bmatrix} \hat{x} - x \\ \nabla_z \log p(z,y) \end{bmatrix} \begin{bmatrix} \hat{x} - x \\ \nabla_z \log p(z,y) \end{bmatrix}^T \right) = \begin{bmatrix} C & M \\ M^T & J \end{bmatrix} \geq 0.$$

A detailed proof is presented in Appendix 4.B.2. □

Since $p(z,y) = p(y \,|\, z)p(z)$, the "information matrix" in the posterior Cramér-Rao bound can be decomposed into two matrices

$$J = J_D + J_P.$$

Above, J_D represents the information in the observed data and J_P the information in the prior,

$$J_D = \mathsf{E}(-\Delta_z^z \log p(y \,|\, z)) \qquad J_P = \mathsf{E}(-\Delta_z^z \log p(z)) \,.$$

Note that J_D is the expectation of the parametric FIM performed under the prior measure $p(z)$. Alternatively, with the decomposition $p(z,y) = p(z \,|\, y)p(y)$, the posterior bound can be written

$$J = \mathsf{E}(-\Delta_z^z \log p(z \,|\, y)) \tag{4.3}$$

since $\Delta_z^z \log p(y) = 0$. Expectation is naturally performed with respect to $p(z,y)$ in (4.3). This expression in some sense justifies the name "posterior" Cramér-Rao bound.

Contrary to the parametric case, the posterior Cramér-Rao bound can be computed even in real applications. Since the parameters are random there is no unknown true parameter value. Instead, in the posterior Cramér-Rao bound the matrices J and M are computed by mathematical expectation performed with respect to the prior distribution of the parameters.

4.2 Discrete Time Nonlinear Estimation

In nonlinear recursive estimation, the primary interest lies in bounds on the filtering or prediction error. For sake of completeness, we will also determine bounds for fixed interval smoothing.

4.2.1 Background

Kerr [100] presents a general review over Cramér-Rao bounds for both continuous and discrete time nonlinear filtering. Although the continuous time case is of less practical interest, it has been studied in much greater detail than the discrete time case. The Cramér-Rao bound for discrete time filtering was initially studied by Borobsky and Zakai [32]. They consider the posterior bound with the restriction to scalar nonlinear models with additive Gaussian noise. Later, Galdos [70] extended the results to the multidimensional case. A more recent generalization of

the same technique is given by Doerschuk [56]. The class of models considered in [56] are discrete time, state space representations of nonlinear autoregressive processes driven by Gaussian state dependent noise having full rank covariance matrix. The approach initiated in [32] and extended in [56, 70] relies on a comparison between the information matrix of the nonlinear system and the information matrix of a suitable linear Gaussian system. According to [100], the argumentation in [32] is somewhat imprecise and lacks a proper definition of the equivalence between the linear and nonlinear models. Some additional limitations of this approach are also discussed in the general overview over lower bounds for nonlinear filtering provided by Kerr [100]. The parametric bound has not been given the same amount of attention in the literature as the posterior bound. A parametric bound is derived by Taylor [144] for the case of continuous time deterministic state evolution and nonlinear discrete time measurements with additive Gaussian noise.

Lately, Tichavský et al. [147] have derived general expressions for posterior Cramér-Rao bounds to discrete time nonlinear filtering using a different, and more general, approach than developed [32]. Similar results were independently obtained by Bergman [17]. Bounds for multidimensional state space models without any limiting Gaussianity assumptions are derived in [147]. The approach to posterior Cramér-Rao bounds for nonlinear filtering advocated by Tichavský et al. has served as an inspiration for the development of a novel parametric Cramér-Rao bound for recursive estimation presented in Section 4.3. The parametric bound can even be evaluated for state space models of greater generality than the ones used for the posterior bound studied in [147]. Due to the lack of prior assumptions, some extra fictitious measurements must be introduced to retain the prediction power of the state space model in the parametric case. Previous work on parametric bounds for discrete time nonlinear filtering utilizes similar techniques to introduce prior information about the system state at the initial time [144]. The parametric bounds presented in Section 4.3 also utilize information about the system state noise and are more general than the one presented in [144]. The case of random parameter estimation is considered in Section 4.4. This section provides a complete proof of the bound for prediction and a review of the filtering bound given by Tichavský et al. [147]. In Section 4.4, we present novel expressions for the posterior Cramér-Rao bound to smoothing problems in nonlinear and possibly non-Gaussian state space models. Finally, in Section 4.5, we give some brief comments on how to practically apply and compute the derived bounds.

4.2.2 Estimation Model

The states x_t are assumed to evolve according to an a priori known, possibly time dependent and nonlinear, discrete time, dynamical state space model

$$\begin{aligned} x_{t+1} &= f_t(x_t, w_t) \\ y_t &= h_t(x_t, e_t) \end{aligned} \qquad t = 0, 1, \ldots \tag{4.4}$$

where $x_t \in \mathbb{R}^n$, $w_t \in \mathbb{R}^m$ and both y_t and e_t are elements of $\mathbb{R}^p$. Let $\{w_t\}$ and $\{e_t\}$ be mutually independent i.i.d. sequences with known densities $p(w_t)$ and $p(e_t)$,

respectively. The initial state is independent of both these noises at all times and has a known probability density function $p(x_0)$. The densities of the state and measurement noises, and the initial state together with the algebraic dependency induced by the model (4.4) determine the stochastic properties of the state and measurement sequences. This gives a correspondence with the generic formulation of recursive estimation problems in Proposition 3.2.

Apart from some regularity conditions such as continuity and differentiability, only one additional restriction is put on the functions in the state space model (4.4). We assume that $h_t(x_t, \cdot) : \mathbb{R}^p \to \mathbb{R}^p$ is bijective for all t and all x_t. Denote the inverse of this function by $h_t^{-1}(x_t, \cdot)$. This assumption is perfectly reasonable from a modeling point of view and it ensures that the likelihood function of y_t given x_t is well defined for all t and x_t,

$$p(y_t \,|\, x_t) = p_{e_t}\left(h_t^{-1}(x_t, y_t)\right). \tag{4.5}$$

The lack of additional restrictions on the state transition equation allows us to include, e.g., the case of models where the conditional state transition density $p(x_{t+1} \,|\, x_t)$ is singular. Such cases must, however, be given some special attention when a random interpretation of the state sequence us utilized. This is studied in the posterior bound case of Section 4.4.

In recursive estimation, the estimator is based on all measurements observed from initial time and the estimate is usually the current or the predicted state vector. Instead of regarding the current state as the parameter and all observed measurements as observations, we study the complete history of states and measurements from the state space model (4.4) and regard the states in past time as *nuisance parameters*. Introduce the vectors of stacked entities from the model (4.4),

$$\mathbb{Y}_t = \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_t \end{bmatrix} \qquad \mathbb{X}_t = \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_t \end{bmatrix} \qquad \begin{aligned} \mathbb{U}_0 &= u_0 = \begin{bmatrix} x_0 \\ w_0 \end{bmatrix} \\ \mathbb{U}_t &= \begin{bmatrix} \mathbb{U}_{t-1} \\ u_t \end{bmatrix} = \begin{bmatrix} \mathbb{U}_{t-1} \\ w_t \end{bmatrix} \quad t > 0 \end{aligned} \tag{4.6}$$

Define the relation from the disturbances driving the model to the current state $x_{t+1} = \varphi_t(\mathbb{U}_t)$ where $\varphi_t : \mathbb{R}^{n+tm} \to \mathbb{R}^n$. We further define the stacked history of such relations $\mathbb{X}_{t+1} = \Phi_t(\mathbb{U}_t)$ where $\Phi_t : \mathbb{R}^{n+tm} \to \mathbb{R}^{(t+1)n}$, i.e.,

$$\mathbb{X}_{t+1} = \Phi_t(\mathbb{U}_t) = \begin{bmatrix} x_0 \\ \varphi_0(\mathbb{U}_0) \\ \varphi_1(\mathbb{U}_1) \\ \vdots \\ \varphi_t(\mathbb{U}_t) \end{bmatrix}. \tag{4.7}$$

In recursive Bayesian estimation of the states of (4.4), the posterior density is one of the marginals of $p(\mathbb{X}_t \,|\, \mathbb{Y}_\tau)$, for different choices of τ. In filtering, it is the case $\tau = t$ that is considered, while prediction treats the case of $\tau < t$, and smoothing

$\tau > t$. When the conditional state transition density, $p(x_{t+1} \mid x_t)$, is singular, the FIM for these estimation problems will be singular as well. While instead treating the vector $\mathbb{U}_t$ as the sought parameter, the FIM for this problem will be invertible and can be used to determine the Cramér-Rao bounds for recursive estimation of the state sequence $\mathbb{X}_t$.

4.3 Parametric Bounds

In the model (4.4), the states are random since both the initial state x_0 and the sequence $\{w_t\}$ are modeled as random vectors. Inserting a specific realization of these random quantities will yield a corresponding realization of the state sequence. This state sequence can be regarded as a fixed but unknown parameter sequence in a parametric framework. In Section 4.3.1 we introduce extra, fictitious, measurements that are used to incorporate prior information about the state sequence into the parametric case.

The parametric Cramér-Rao bounds for recursive estimation are derived in Section 4.3.2. The resulting bounds are found to coincide with the error covariance Riccati recursion of the Kalman filter to the system obtained when linearizing the model around the true state sequence. Parametric bounds for filtering, prediction, and smoothing are presented with derivations in Section 4.3.2.

4.3.1 Fictitious Measurements

Denote the fixed but unknown true sequence of states by $\mathbb{X}_t^\star$. This vector can thus alternatively be specified by $\mathbb{U}_{t-1}^\star$ since the state transition part of the model yields that $\Phi_{t-1}(\mathbb{U}_{t-1}^\star) = \mathbb{X}_t^\star$. Given the measurements $\mathbb{Y}_t$ and a pure parametric view on the state sequence, very little can be said about future state values $x_{t+\tau}$. Since no prior information is available about the unknown future values of w_t, the limitations induced by the state space model are strictly algebraic, and lower bounds for parametric prediction of the state sequence will therefore in general be impractically high. The unknown but fixed parameter sequence $\mathbb{X}_t^\star$ must naturally satisfy the model

$$x_{t+1}^\star = f_t(x_t^\star, w_t^\star) \qquad t = 0, 1, \ldots,$$

but the future true parameters $w_t^\star$ are unknown, and the only knowledge about them is that they are elements in $\mathbb{R}^m$. The range of possible future states is therefore in general unlimited and the measurements $\mathbb{Y}_t$ carry no information about these state values.

However, the prediction power inherent in the state space model (4.4) can be retained by introducing some regularizing prior information about the parameters $w_t^\star$. Assume that we are provided with some average value of $w_t^\star$, and a probabilistic model for the difference between the true parameter and this average value. That is, let

$$w_t^\star = \bar{w}_t + \tilde{w}_t$$

where $\bar{w}_t$ is the average value of the unknown parameter and $\tilde{w}_t$ is a random, zero mean, error having well a defined density $p_{\tilde{w}_t}(\tilde{w}_t)$. The average value $\bar{w}_t$ is regarded as a measurement of the parameter $w_t^\star$. This measurement and the density function $p_{\tilde{w}_t}(\tilde{w}_t)$ defines the prior knowledge that retains the prediction ability in the parametric framework. Similar prior knowledge is also needed for the initial state of the model (4.4). Thus, we define some additional measurements, z_t, originating from a measurement model with

$$\begin{aligned} \boldsymbol{z}_0 &= \begin{bmatrix} x_0 - \tilde{\boldsymbol{x}}_0 \\ w_0 - \tilde{\boldsymbol{w}}_0 \end{bmatrix} \qquad \text{and} \\ \boldsymbol{z}_t &= w_t - \tilde{\boldsymbol{w}}_t \qquad\quad \text{for } t = 1, 2, \ldots \end{aligned} \tag{4.8}$$

In (4.8) we clearly indicate which quantities are stochastic and which are deterministic but this notational distinction is not pursued in the sequel. The observed event at initial time is $z_0 = \left[\begin{smallmatrix} \bar{x}_0 \\ \bar{w}_0 \end{smallmatrix}\right]$ and at later times $z_t = \bar{w}_t$. We also define $\mathbb{Z}_t$ to be the stacked vector of the measurements z_t,

$$\mathbb{Z}_t^T = \begin{bmatrix} z_0^T & z_1^T & \ldots & z_t^T \end{bmatrix}. \tag{4.9}$$

The likelihood for the fictitious measurements (4.8) is thus given by

$$\begin{aligned} p(z_0 = \left[\begin{smallmatrix} \bar{x}_0 \\ \bar{w}_0 \end{smallmatrix}\right] \,|\, x_0, w_0) &= p_{\tilde{x}_0}(x_0 - \bar{x}_0) p_{\tilde{w}_0}(w_0 - \bar{w}_0) \\ p(z_t = \bar{w}_t \,|\, w_t) &= p_{\tilde{w}_t}(w_t - \bar{w}_t) \end{aligned}$$

where the interpretation of the measurements as the average value of the parameter is obvious. Without loss of generality we can, for the sake of the Cramér-Rao bound calculations, assume that all of $\bar{w}_t$ and $\bar{x}_0$ are equal to zero, and for notational convenience set $p_{\tilde{x}_0}(\cdot) = p_{x_0}(\cdot)$, and $p_{\tilde{w}_t}(\cdot) = p_{w_t}(\cdot)$. The likelihood of the fictitious measurements will then have the exact role of the prior distributions in the Bayesian framework,

$$p(z_0 = 0 \,|\, x_0, w_0) = p_{x_0}(x_0) p_{w_0}(w_0) \qquad p(z_t = 0 \,|\, w_t) = p_{w_t}(w_t).$$

This yields a more compact notation and makes comparisons with the random case easier. In previous work on parametric Cramér-Rao bounds for nonlinear filtering, Taylor [144] also introduces the concept of an additional measurement for the prior information about the initial state. This introduction of fictitious measurements corresponds to real measurements in the terrain navigation application. The heading and speed estimates from the inertial navigation system are measurements that are used to determine the mean value of the system noise. The variance of these measurements is prior information about the drift of the inertial system during one sample interval, as described in Chapter 2. When estimation of $\mathbb{X}_t$, or specifically x_t, is considered, it is in the sequel understood that the measurements $\mathbb{Z}_{t-1}$ have been observed as an all zero vector. Since the bounds are derived using these fictitious measurements, the estimators evaluated against the bounds may also have the ability to utilize the information in the measurements (4.8). In practice, this

is obtained by introducing prior information, e.g., in the way used in the terrain navigation application. This will also be further exemplified in the Cramér-Rao bound evaluation in Chapter 7.

The fictitious measurements are introduced solely since the lack of prior information about the states seriously degrades the prediction power of the state space model. For filtering and smoothing, we are not obliged to use the additional measurements. A rigorous derivation of expressions for the parametric Cramér-Rao bound for filtering and smoothing without the additional measurements can be performed in a straightforward manner. We refrain from this option since it builds on a very similar argumentation to the derivation presented below, with the measurements incorporated into the Cramér-Rao bound. Moreover, the state evolution equation will only play a minor role for the bound if no prior information about the state noise parameters w_t is given. Consider, e.g., the case of additive noise in the state transition,

$$x_{t+1} = f_t(x_t) + w_t. \tag{4.10a}$$

With no prior information about w_t, the state evolution equation will not affect the bound at all since the value of x_t says nothing about x_{t+1}. In this case, the model simplifies to a series of nonlinear regressions affected by white noise measurement error

$$y_t = h_t(x_t, e_t) \tag{4.10b}$$

where each x_t is a completely new parameter without statistical connection to any other x_τ, $\tau \neq t$. Hence, in a parametric framework without any prior information about w_t, the model (4.10) does not correspond to a filtering problem since there is no dynamical dependency in the state sequence.

4.3.2 Bound Calculations

Consider the parametric inference problem of estimating the fixed state sequence $\mathbb{X}_{t+1}^\star$, where $\mathbb{X}_{t+1}^\star = \Phi_t(\mathbb{U}_t^\star)$ according to (4.7). The estimate $\hat{\mathbb{X}}_{t+1}$ is based on the sequence $\mathbb{Y}_t$, given in (4.6) and the set of fictitious measurements $\mathbb{Z}_t$, from (4.7). The Cramér-Rao bound states that the mean square error of any unbiased estimator $\hat{\mathbb{X}}_{t+1}$ satisfies

$$\mathsf{E}\Big((\hat{\mathbb{X}}_{t+1} - \mathbb{X}_{t+1}^\star)(\hat{\mathbb{X}}_{t+1} - \mathbb{X}_{t+1}^\star)^T\Big) \geq \mathbb{P}_{t+1}, \tag{4.11}$$

where, according to Theorem 4.1, the lower bound is given by

$$\mathbb{P}_{t+1} = \left(\nabla_{\mathbb{U}_t} \Phi_t^T(\mathbb{U}_t)\right)^T J_t^{-1} \left(\nabla_{\mathbb{U}_t} \Phi_t^T(\mathbb{U}_t)\right)\Big|_{\mathbb{U}_t = \mathbb{U}_t^\star} \tag{4.12a}$$

$$J_t = \mathsf{E}\Big(-\Delta_{\mathbb{U}_t}^{\mathbb{U}_t} \log p(\mathbb{Y}_t, \mathbb{Z}_t \,|\, \mathbb{U}_t)\Big)\Big|_{\mathbb{U}_t = \mathbb{U}_t^\star} \tag{4.12b}$$

and expectation is performed with respect to the likelihood $p(\mathbb{Y}_t, \mathbb{Z}_t \,|\, \mathbb{U}_t)$. Let P_{t+1} denote the Cramér-Rao bound for the one step ahead prediction of the state x_{t+1} given measurements $\mathbb{Y}_t$ and the prior knowledge defined by observing the all zero vector $\mathbb{Z}_t$. By decomposing the lower bound (4.11) into sub-blocks, the Cramér-Rao bound for this prediction problem is found in the lower right corner of $\mathbb{P}_{t+1}$,

$$\mathsf{E}\left(\begin{bmatrix} \hat{\mathbb{X}}_t - \mathbb{X}_t^\star \\ \hat{x}_{t+1} - x_{t+1}^\star \end{bmatrix} \begin{bmatrix} \hat{\mathbb{X}}_t - \mathbb{X}_t^\star \\ \hat{x}_{t+1} - x_{t+1}^\star \end{bmatrix}^T \right) \geq \begin{bmatrix} * & * \\ * & P_{t+1} \end{bmatrix}. \tag{4.13}$$

The historical part of the state sequence $\mathbb{X}_t$ is here regarded as *nuisance parameters*. The parametric Cramér-Rao bound for both the filtering and the smoothing problems can similarly be found on the block diagonal of the matrix $\mathbb{P}_{t+1}$. Straightforwardly computing $\mathbb{P}_{t+1}$ by forming (4.12) and extracting these block diagonal elements would require increasingly tedious computations. The sought block diagonal elements can instead be expressed recursively by exploiting the structure of the Cramér-Rao bound matrix $\mathbb{P}_{t+1}$.

Theorem 4.3 (Parametric Prediction)
The Cramér-Rao bound for the problem of one step ahead prediction of the unknown but fixed states of (4.4) *is*

$$\mathsf{E}\big((\hat{x}_{t+1}(\mathbb{Y}_t, \mathbb{Z}_t) - x_{t+1}^\star)(\hat{x}_{t+1}(\mathbb{Y}_t, \mathbb{Z}_t) - x_{t+1}^\star)^T\big) \geq P_{t+1}$$

where $P_0^{-1} = -\Delta_{x_0}^{x_0} \log p_{x_0}(x_0^\star)$, *and*

$$P_{t+1} = F_t \left(P_t - P_t H_t^T (H_t P_t H_t^T + R_t)^{-1} H_t P_t\right) F_t^T + G_t Q_t G_t^T.$$

The matrices F_t *and* G_t *are given by gradients of the functions defining the model* (4.4), *evaluated at the true trajectory,*

$$F_t^T = \nabla_{x_t} f_t^T(x_t^\star, w_t^\star) \qquad\qquad G_t^T = \nabla_{w_t} f_t^T(x_t^\star, w_t^\star).$$

The matrices H_t, R_t *and* Q_t *are similarly given by partial derivatives of the model expressions,*

$$H_t^T R_t^{-1} H_t = \mathsf{E}\big(-\Delta_{x_t}^{x_t} p(y_t \,|\, x_t^\star)\big) \qquad\qquad Q_t^{-1} = \mathsf{E}\big(-\Delta_{w_t}^{w_t} p(z_t \,|\, w_t^\star)\big)$$

where the factorization between H_t *and* R_t *is such that* $R_t > 0$ *and expectation is performed with respect to* $p(y_t \,|\, x_t^\star)$ *and* $p(z_t \,|\, w_t^\star)$, *respectively. The likelihood* $p(y_t \,|\, x_t)$ *is given in* (4.5). *The Cramér-Rao bound inequality is valid under the assumption that the abovementioned gradients, expectations and matrix inverses exist.*

Proof See Appendix 4.C.1. □

The expression for the parametric Cramér-Rao bound P_t is recognized as the Riccati recursion for the error covariance in a Kalman filter, see Theorem 3.4. It is

straightforward to incorporate the case of statistically dependent observation and state transition noise into the Cramér-Rao bound above. This yields a perfect resemblance with Theorem 3.4. Thus, the bound P_t is the error covariance of the Extended Kalman Filter (EKF), obtained by linearizing the state equations around the true state trajectory. Thus, the EKF would be an efficient estimator if this linearization could be performed. However, this would make it unnecessary to perform any estimation since it would require that the true trajectory was known.

The theorem gives a recursive expression for the bound where the recursive update is determined by gradients and expectations over the model functions and densities. In Section 4.5, we will comment on the issue of computing these expectations whenever they lack known analytical solutions. In practice, one must also consider the implicit definition of the matrices H_t and R_t in Theorem 4.3. The factorization between H_t and R_t in the theorem will depend on the actual model studied. Generally, it can be computed by singular value decomposition once the matrix $\mathsf{E}\big(-\Delta_{x_t}^{x_t} \log p(y_t \,|\, x_t)\big)$ has been formed, or possibly by an analytical investigation into the measurement model of the problem. Often, the choice

$$H_t^T = \nabla_{x_t} \left(h_t^{-1}(x_t, y_t) \right)^T$$

will suffice, at least if this gradient is independent of y_t. This will, e.g., be the case when the noise enters the measurement equation additively. If, additionally, the noise distribution is Gaussian, the expectations in the bound recursion can be analytically evaluated.

Corollary 4.2 (Additive Gaussian Measurement Noise)
Consider the prediction problem of Theorem 4.3. Under the assumption that the measurement noise enters additively,

$$y_t = h_t(x_t) + e_t$$

and that $p(x_0)$, $p(w_t)$ and $p(e_t)$ are Gaussian distributed with positive definite covariance matrices P_0, Q_t and R_t, respectively, the Cramér-Rao bound is given by the recursion

$$P_{t+1} = F_t \left(P_t - P_t H_t^T (H_t P_t H_t^T + R_t)^{-1} H_t P_t \right) F_t^T + G_t Q_t G_t^T.$$

The matrices F_t and G_t are given in Theorem 4.3, while

$$H_t^T = \nabla_{x_t} h_t^T(x_t).$$

Proof Any Gaussian density with positive definite covariance satisfies

$$-\Delta_x^x \log \mathrm{N}(x, ; \mu, P) = P^{-1}.$$

The likelihood $p(y_t \,|\, x_t) = p_{e_t}(y_t - h_t(x_t))$ yields that

$$\mathsf{E}(-\Delta_{x_t}^{x_t} \log p(y_t \,|\, x_t)) = \nabla_{x_t} h_t^T(x_t) R_t^{-1} \left(\nabla_{x_t} h_t^T(x_t) \right)^T$$

since expectation is performed with respect to y_t. □

The Cramér-Rao bound for filtering can be found using a partitioning, similar to the one in (4.13)

$$\mathsf{E}\left(\begin{bmatrix}\hat{\mathbb{X}}_{t-1}-\mathbb{X}^\star_{t-1}\\ \hat{x}_t - x_t^\star \\ \hat{x}_{t+1}-x^\star_{t+1}\end{bmatrix}\begin{bmatrix}\hat{\mathbb{X}}_{t-1}-\mathbb{X}^\star_{t-1}\\ \hat{x}_t - x_t^\star \\ \hat{x}_{t+1}-x^\star_{t+1}\end{bmatrix}^T\right) \geq \begin{bmatrix} * & * & * \\ * & P_{t|t} & * \\ * & * & P_{t+1}\end{bmatrix}$$

Utilizing the analogy between the Cramér-Rao bound of Theorem 4.3 and the Kalman filter recursion, the bound recursion can be decomposed into a time update and a measurement update step. The Cramér-Rao bound for the filtering problem thus follows by performing these steps in opposite order, and initiating the recursion with one additional measurement update.

Theorem 4.4 (Parametric Filtering)
The Cramér-Rao bound for the filtering of the states of (4.4) *is given by*

$$\mathsf{E}\big((\hat{x}_t(\mathbb{Y}_t,\mathbb{Z}_{t-1})-x_t^\star)(\hat{x}_t(\mathbb{Y}_t,\mathbb{Z}_{t-1})-x_t^\star)^T\big) \geq P_{t|t}$$

where $P_{0|0} = P_0 - P_0 H_0^T \left(H_0 P_0 H_0^T + R_0\right)^{-1} H_0 P_0$ *and*

$$\begin{aligned} P_{t+1|t} &= F_t P_{t|t} F_t^T + G_t Q_t G_t^T \\ P_{t+1|t+1} &= P_{t+1|t} - P_{t+1|t} H_t^T \left(H_t P_{t+1|t} H_t^T + R_t\right)^{-1} H_t P_{t+1|t}. \end{aligned}$$

The matrices F_t, G_t, H_t, Q_t *and* R_t *are defined in Theorem 4.3.*

Proof Follows by minor adjustments in the proof of Theorem 4.3. □

The bound for the smoothing problem also follows from the Kalman filter analogy. The bound results in the usual type of forward-backward recursions for fixed interval smoothing. The smoothing case is considered in greater detail for the posterior case in the following section.

4.4 Posterior Bounds

For the random case, it is the joint density of parameters and observations that enters the Cramér-Rao bound calculations. Using the notation defined in (4.6), the joint density for the one step ahead prediction problem is

$$p(\mathbb{Y}_t, \mathbb{U}_t) = p(y_t \,|\, \mathbb{U}_{t-1}) p(w_t) p(\mathbb{Y}_{t-1}, \mathbb{U}_{t-1}).$$

However, applying a similar approach as the one in the parametric case is not possible for the posterior bound. Following the notation in the proof of Theorem 4.3, the

matrices for the Cramér-Rao bound can be decomposed and recursively expressed as

$$J_t = \mathsf{E}\left(\begin{bmatrix} W_{t-1}^T H_t^T R_t^{-1} H_t W_{t-1} & 0 \\ 0 & Q_t^{-1} \end{bmatrix} \right) + \begin{bmatrix} J_{t-1} & 0 \\ 0 & 0 \end{bmatrix} \tag{4.14a}$$

$$M_t = \mathsf{E}\left(\begin{bmatrix} 0 & 0 \\ F_t W_{t-1} & G_t \end{bmatrix} \right) + \begin{bmatrix} M_{t-1} & 0 \\ 0 & 0 \end{bmatrix} \tag{4.14b}$$

where all the matrices inside the expectation operator depend on either or both of the random quantities $\mathbb{U}_t$ and y_t. The expectation should be performed over both $\mathbb{U}_t$ and y_t using the joint density $p(y_t, \mathbb{U}_t)$. These expectations hinder us from obtaining a recursion similar to the one for the parametric case. Further restrictions need to be introduced into the model (4.4) in order to obtain posterior bounds with limited computational complexity.

Tichavský et al. [147] introduce the additional assumption that the transition kernel $p(x_{t+1} \mid x_t)$ is a well defined probability density function for x_{t+1} at all times t and given any $x_t \in \mathbb{R}^n$. Specifically, the function $p(x_{t+1} \mid x_t)$ must be twice differentiable with respect to both x_t and x_{t+1}. A necessary condition for this assumption to hold is that the system noise $w_t \in \mathbb{R}^n$. This is commonly not the case, e.g., in target tracking applications. The bound for nonlinear filtering is considered in [147]. We present both cases of prediction and filtering in a single theorem.

Theorem 4.5 (Posterior filtering and prediction bounds)
The posterior Cramér-Rao bound for the one step ahead prediction of the states of (4.4) *is given by the recursion*

$$P_{t+1}^{-1} = Q_t - S_t^T \left(P_t^{-1} + R_t + V_t \right)^{-1} S_t$$

initiated by $P_0^{-1} = \mathsf{E}\big(-\Delta_{x_0}^{x_0} \log p(x_0)\big)$, *and where*

$$V_t = \mathsf{E}\big(-\Delta_{x_t}^{x_t} \log p(x_{t+1} \mid x_t)\big) \qquad R_t = \mathsf{E}\big(-\Delta_{x_t}^{x_t} \log p(y_t \mid x_t)\big)$$
$$S_t = \mathsf{E}\big(-\Delta_{x_t}^{x_{t+1}} \log p(x_{t+1} \mid x_t)\big) \qquad Q_t = \mathsf{E}\Big(-\Delta_{x_{t+1}}^{x_{t+1}} \log p(x_{t+1} \mid x_t)\Big).$$

The filtering case is initiated by $P_{0|0}^{-1} = (P_0^{-1} + R_0)^{-1}$ *and recursively determined as*

$$P_{t+1|t+1}^{-1} = Q_t + R_{t+1} - S_t^T \left(P_{t|t}^{-1} + V_t \right)^{-1} S_t.$$

The expressions hold under the assumption that the involved differentiations and expectations exist. Explicitly, this assumes that $p(x_{t+1} \mid x_t)$ *is well defined for all* t.

Proof See Appendix 4.C.2 for a proof of the prediction bound. The filtering case follows similarly, or alternatively refer to [147], or [146]. The bound for prediction will soon also appear in [134]. □

For the linear Gaussian case, the recursive expressions for the bounds will once again coincide with the Kalman filter recursions. Generally, the likelihood $p(y_t \,|\, x_t)$ is given by (4.5) and the state transition kernel $p(x_{t+1} \,|\, x_t)$ implicitly defined by (4.4). To be more precise, consider nonlinear models with additive noise

$$\begin{aligned} x_{t+1} &= f_t(x_t) + w_t \\ y_t &= h_t(x_t) + e_t \end{aligned} \qquad t = 0, 1, \ldots \tag{4.15}$$

where we assume that the noise w_t, e_t, and the initial state x_0 all are Gaussian distributed with known positive definite covariances $\widetilde{Q}_t$, $\widetilde{R}_t$, and P_0, respectively. This model is covered by Theorem 4.5 and follows in a straightforward way by verifying that

$$\begin{aligned} V_t &= \mathsf{E}\Big(\nabla_{x_t} f_t^T(x_t) \widetilde{Q}_t^{-1} (\nabla_{x_t} f_t^T(x_t))^T\Big) & S_t &= -\mathsf{E}\big(\nabla_{x_t} f_t^T(x_t)\big)\, \widetilde{Q}_t^{-1} \\ R_t &= \mathsf{E}\Big(\nabla_{x_t} h_t^T(x_t) \widetilde{R}_t^{-1} (\nabla_{x_t} h_t^T(x_t))^T\Big) & Q_t &= \widetilde{Q}_t^{-1} \end{aligned}$$

which defines the recursion for the bound defined in Theorem 4.5.

The main restrictive assumption of Theorem 4.2 is the regularity assumption on the transition kernel $p(x_{t+1} \,|\, x_t)$. Tichavský et al. [147] suggest to handle the cases when $p(x_{t+1} \,|\, x_t)$ is singular by adding extra "regularization noise" with small covariance in the state transition equation. However, the bound for the case with singular transition kernel $p(x_{t+1} \,|\, x_t)$ can actually be computed without some additional regularization noise for models with linear state evolution but possibly nonlinear measurement relation. Consider the estimation model

$$\begin{aligned} x_{t+1} &= F_t x_t + G_t w_t \\ y_t &= h_t(x_t, e_t) \end{aligned} \qquad t = 0, 1, \ldots \tag{4.16}$$

where $w_t \in \mathbb{R}^m$ and $m \leq n$. As usual, the densities $p(x_0)$, $p(w_t)$, and $p(e_t)$ are assumed known, but otherwise not restricted to any special class of distributions. This class of models, although severely more restrictive than (4.4), still captures many practical estimation problems. Several target tracking applications can actually be written in the form given in (4.16).

Theorem 4.6 (Posterior bound with singular state evolution)
The posterior Cramér-Rao bound for the one step ahead prediction of the states of (4.16) *is given by the recursion*

$$P_{t+1} = F_t \left(P_t - P_t (P_t + Z_t^{-1})^{-1} P_t\right) F_t^T + G_t Q_t G_t^T.$$

The recursion is initiated by

$$P_0^{-1} = \mathsf{E}\big(-\Delta_{x_0}^{x_0} \log p_{x_0}(x_0)\big)$$

and

$$Q_t^{-1} = \mathsf{E}\big(-\Delta_{w_t}^{w_t} \log p_{w_t}(w_t)\big) \qquad Z_t = \mathsf{E}\big(-\Delta_{x_t}^{x_t} \log p(y_t \,|\, x_t)\big)$$

where the likelihood is given in (4.5)*. The expectations above are performed with respect to* x_0*,* w_t*, and the joint density* $p(y_t, \mathbb{U}_{t-1})$*, respectively. The result holds under the assumption that the expectations, differentiations and matrix inverses above exist.*

Proof Since the relation $x_{t+1} = \varphi_t(\mathbb{U}_t)$ becomes linear for the model (4.16), $x_{t+1} = W_t\mathbb{U}_t$ and the gradient $\nabla_{\mathbb{U}_t} x_{t+1}^T = W_t^T$ is a constant matrix which is unaffected by the expectations in (4.14). Thus, a technique similar to the one in Appendix 4.C.1 is applicable to the model class of this theorem. A detailed proof is given in Appendix 4.C.3 for sake of completeness. □

Corresponding filtering and smoothing formulas for the case in Theorem 4.6 follow similarly as in the parametric case. Instead, we consider the bound for smoothing under the assumption that $p(x_{t+1} \,|\, x_t)$ is nonsingular. This bound follows from a similar technique as the one used in Theorem 4.5 and the bounds presented by Tichavský et al. [147], but the smoothing bounds are not considered by Tichavský et al. [147].

Theorem 4.7 (Posterior Cramér-Rao bound for smoothing) *Under the assumption that* $p(x_{t+1} \,|\, x_t)$ *is nonsingular, the posterior Cramér-Rao bound for smoothing of the states in* (4.4)

$$\mathsf{E}\big((\hat{x}_t(\mathbb{Y}_N) - x_t)(\hat{x}_t(\mathbb{Y}_N) - x_t)^T\big) \geq P_{t|N}$$

can be computed using a forward-backward recursion. Initiated by

$$P_0^{-1} = \mathsf{E}\big(-\Delta_{x_0}^{x_0} \log p(x_0)\big)\,,$$

the forward recursion is

$$P_{t+1}^{-1} = Q_t - S_t^T \left(P_t^{-1} + R_t + V_t\right)^{-1} S_t \qquad t = 0, 1, \ldots, N$$

Initiated by $P_{N+1|N} = P_{N+1}$*, the backward recursion becomes*

$$P_{t|N} = P_{t|t} + P_{t|t} S_t P_{t+1|N} S_t^T P_{t|t} \qquad t = N, N-1, \ldots, 0$$

where

$$P_{t|t} = \left(R_t + V_t + P_t^{-1}\right)^{-1}.$$

The matrices Q_t*,* R_t*,* S_t *and* V_t *are defined in Theorem 4.5.*

Proof See Appendix 4.C.4. □

The theorem presents the bound for fixed interval smoothing, fixed lag or point smoothing follows similarly. For the linear Gaussian case, the recursions above will coincide with the Rauch-Tung-Striebel (RTS) formulas for optimal linear smoothing [2, 96].

4.5 Relative Monte Carlo Evaluation

There are several ways to utilize the bounds for recursive estimation presented in this chapter. Lower bounds on estimation performance can, e.g., be used to determine whether the imposed design specifications are realistic. Alternatively, plotting the bound versus the range of different unknown but not estimated model parameters, e.g., versus the signal to noise ratio, can give valuable insight into the character of the estimation problem. In this work we utilize the bounds for evaluation of suboptimal algorithms by means of Monte Carlo simulations. Such an evaluation can be used to determine the suboptimal performance of the algorithm, but also used to verify the correctness of the actual implementation of an optimal algorithm.

Comparison of the performance of several algorithms on the basis of their Monte Carlo Root Mean Square (RMS) error is commonly utilized in signal processing applications. A relative comparison between different algorithms can be used to choose the most favorable approach for the problem at hand. The absolute RMS error, on the other hand, only reveals the average performance of the algorithm under the simulation conditions used in the Monte Carlo evaluation. This error value will naturally depend on the subjective choice of noise levels and distributions used in the particular simulation study. By computing a lower bound to compare the results with, a measure of the relative effectiveness of each algorithm is obtained. The discrepancy of the absolute RMS error from this lower bound reveals the effects of any suboptimal approximations introduced in the current algorithm.

In nonlinear estimation, the RMS error may depend on the region of the state space that the simulation is performed over. This is the case in terrain navigation where simulations over flat terrain usually yields higher RMS errors than identical simulations performed over rough terrain. Examples of such behavior are given both in Chapter 2, and Sections 7.2 and 7.3. A statement about the average position error of a terrain navigation algorithm in a simulation study therefore says nothing about the general performance of the algorithm. The statement needs to be accompanied by a measure of the terrain variation in the area where the simulations were performed. The Cramér-Rao bound can be seen as a very natural way to quantize the terrain information. The relative difference between the bound and the algorithm performance is a measure that makes sense even if two different algorithms are evaluated over different terrain areas. Hence, the lower bound yields an opportunity to compare the Monte Carlo RMS performance between algorithms tested under different circumstances.

Consider the parametric one step ahead prediction problem, with a Cramér-Rao bound given by the matrix inequality

$$\mathsf{E}\big((\hat{x}_t - x_t^\star)(\hat{x}_t - x_t^\star)^T\big) \geq P_t, \tag{4.17}$$

where P_t is recursively defined by Theorem 4.3. This matrix inequality is hard to depict, at least if the state space is of dimension greater than two. Consider instead

the scalar mean square error given by the trace of (4.17),

$$\mathsf{E}\big((\hat{x}_t - x_t^\star)^T(\hat{x}_t - x_t^\star)\big) = \mathsf{E}\Big(\|\hat{x}_t - x_t^\star\|^2\Big) \geq \operatorname{tr} P_t. \tag{4.18}$$

Both the left hand side mean square error and possibly the right hand side Cramér-Rao bound recursion involve expectations which can be estimated using Monte Carlo simulations.

In a simulation study under the parametric paradigm, a true trajectory $\mathbb{X}_t^\star$ of length N must be determined. This fixed state trajectory can be generated by choosing $x_0^\star$ and $w_t^\star$ and then running these parameters through the state space model, i.e., computing $\mathbb{X}_t^\star = \Phi_{t-1}(\mathbb{U}_t^\star)$. The algorithm that should be evaluated is then applied M times to this trajectory using different, but identically distributed, realizations of the measurement noise. Let $\hat{x}_t^i$ denote the estimate produced by the algorithm at time t in Monte Carlo iteration i. Since the expectation in (4.18) is performed over the measurement distribution, this expectation can be estimated by a sample mean over the independent measurement realizations. This is an application of the principle of Monte Carlo integration which will be thoroughly discussed and applied to recursive estimation in Chapter 6. Inserting the Monte Carlo estimate and taking the square root on both sides of (4.18) yields that

$$\sqrt{\frac{1}{M}\sum_{i=1}^{M} \|\hat{x}_t^i - x_t^\star\|^2} \gtrsim \sqrt{\operatorname{tr} P_t} \qquad t = 0, 1, \ldots, N$$

where the notation $\gtrsim$ is used to denote that the inequality only holds approximately for finite M. Hence, the simulation root mean square (RMS) error in a Monte Carlo evaluation is bounded from below by the square root of the trace of the Cramér-Rao bound. The Cramér-Rao bound is computed as in Theorem 4.3 where the linearizations around the true state trajectory are computed analytically or numerically. The expectations for evaluating R_t and Q_t for Theorem 4.3 can also be computed by Monte Carlo techniques or evaluated analytically, e.g., if the noises are Gaussian distributed.

The same Monte Carlo approach can be used in the posterior case. The only difference is that the RMS error should be averaged over M *different* state trajectories, with different measurement noise independently generated according to the system model. The Monte Carlo limit for the posterior case thus becomes

$$\sqrt{\frac{1}{M}\sum_{i=1}^{M} \|\hat{x}_t^i - x_t^i\|^2} \gtrsim \sqrt{\operatorname{tr} P_t} \qquad t = 0, 1, \ldots, N$$

where the right hand side is given by Theorem 4.5, and x_t^i is the true system state at time t in Monte Carlo run i. The matrices in the posterior Cramér-Rao bound recursion involve expectations over the joint density. These expectations can be computed using a similar Monte Carlo approach. In the posterior bound of Theorem 4.5 the matrix

$$R_t = \mathsf{E}\big(-\Delta_{x_t}^{x_t} \log p(y_t \,|\, x_t)\big)$$

is needed for the recursion. With additive Gaussian measurement noise having positive definite covariance $\widetilde{R}_t$, the estimate

$$\widehat{R}_t \approx \frac{1}{M} \sum_{i=1}^{M} \nabla_{x_t} h_t^T(x_t^i) \widetilde{R}_t^{-1} \left(\nabla_{x_t} h_t^T(x_t^i) \right)^T$$

is an appropriate estimate of R_t. The number M must be chosen sufficiently large, since both the algorithm RMS error and the Cramér-Rao bound increase in reliability with the number of Monte Carlo iterations.

Estimators with a mean square error that reaches the Cramér-Rao bound are often referred to as *efficient* estimators. A suboptimal algorithm for which the Monte Carlo RMS error stays in the vicinity of the bound will naturally imply that the algorithm is good. However, if the RMS error does not reach the bound this does not necessarily imply that it is a bad algorithm. There might not exist any efficient estimator for the current estimation problem [151]. In the parametric case, it is well known that if an efficient estimator exists, the maximum likelihood estimator will be efficient and have mean square error equal to the Cramér-Rao bound [151]. On the other hand, one may also construct estimators for which the mean square error falls below the Cramér-Rao bound. Such *superefficiency* usually originates from a bias variance trade-off for a finite number of observations, but can actually also occur asymptotically. Stoica and Ottersten [139] present a general review of the theoretical background to such behavior exemplified by some, non-trivial yet simple, superefficient estimators.

4.6 Conclusions

Expressions for several Cramér-Rao bounds have been presented in this chapter. Bounds for filtering, prediction and smoothing have been given both for the parametric and the random view on nonlinear state space estimation. Table 4.1 presents a quick overview of the different expressions for Cramér-Rao bounds derived in this chapter. The expressions for the bounds are given as recursive formulas for rather general classes of state space estimation models. The recursive expressions involve differentiations of the model equations and expectations over these. Both the bound and the algorithm performance can straightforwardly be evaluated in standard Monte Carlo simulations. By performing these Monte Carlo simulations under simplified conditions, e.g., assuming additive Gaussian noises, the bound can often be explicitly determined. Such simulations must naturally be accompanied by more realistic simulations for evaluation of the absolute performance of the algorithm. In Chapter 7 we present extensive Monte Carlo simulations using both the parametric and the posterior bounds to evaluate an approximative algorithm for terrain navigation.

Theorem 4.1, page 53	Bound for general unbiased estimators in parametric inference, $C \geq MJ^{-1}M^T$.
Corollary 4.1, page 54	Bound for parametric estimators with unknown bias.
Theorem 4.2, page 55	Bound for general random inference, $C \geq MJ^{-1}M^T$.
Theorem 4.3, page 62	Parametric one step ahead prediction using the general model (4.4) and some fictitious measurements.
Theorem 4.4, page 64	Parametric filtering for the abovementioned case.
Theorem 4.5, page 65	Posterior bounds for filtering and prediction under the assumption that $p(x_{t+1} \mid x_t)$ is nonsingular.
Theorem 4.6, page 66	Posterior bound for possibly singular but linear state evolution.
Theorem 4.7, page 67	Posterior bound for smoothing under the assumption that $p(x_{t+1} \mid x_t)$ is nonsingular.

Table 4.1: Summary of the different results on Cramér-Rao bounds.

Appendix

4.A Vector Gradients and Matrix Inversions

The presentation in this chapter utilizes vector gradient and block matrix notation. For reader's convenience, we summarize some of the basic definitions and results regarding these matters in this appendix. This first subsection presents the definition of vector gradients and some expressions involving such operators. The second subsection presents two commonly applied matrix inversion formulas. The expressions have been borrowed from Graham [79] and Kailath [95], respectively. Several other useful relations can be found in these references.

4.A.1 Vector Gradients

Let $x \in \mathbb{R}^p$, and $a(x) \mapsto \mathbb{R}$ be a scalar valued function of x. The gradient of $a(x)$ with respect to x is the $p \times 1$ vector

$$\nabla_x a(x) \triangleq \begin{bmatrix} \frac{\partial a(x)}{\partial x_1} \\ \frac{\partial a(x)}{\partial x_2} \\ \vdots \\ \frac{\partial a(x)}{\partial x_p} \end{bmatrix}.$$

The gradient operator ∇_x is a column vector of scalar partial derivative operators. The generalization to vector valued functions therefore follows by standard matrix multiplication. Thus, let $a(x) \mapsto \mathbb{R}^n$, be a vector valued function of $x \in \mathbb{R}^p$, then

$$\nabla_x a^T(x) = \begin{bmatrix} \frac{\partial a_1(x)}{\partial x_1} & \cdots & \frac{\partial a_n(x)}{\partial x_1} \\ \vdots & & \vdots \\ \frac{\partial a_1(x)}{\partial x_p} & \cdots & \frac{\partial a_n(x)}{\partial x_p} \end{bmatrix}.$$

Some useful relations follow straightforwardly from the definitions above

$$\nabla_x x^T = I \qquad \nabla_x v^T x = \nabla_x x^T v = v$$

$$\nabla_x a^T(x)b(x) = \Big(\nabla_x a^T(x)\Big)b(x) + \Big(\nabla_x b^T(x)\Big)a(x)$$
$$\nabla_x x^T M x = (M + M^T)x$$

if v and M are constant, e.g., independent of x.

Let $a(x, y) \mapsto \mathbb{R}$ be a scalar valued function of two vectors, $x \in \mathbb{R}^p$ and $y \in \mathbb{R}^r$. The Laplacian of $a(x, y)$ with respect to x and y is the $r \times p$ matrix

$$\Delta_y^x a(x,y) \triangleq \nabla_y \nabla_x^T a(x,y) = \begin{bmatrix} \frac{\partial^2 a(x,y)}{\partial y_1 \partial x_1} & \cdots & \frac{\partial^2 a(x,y)}{\partial y_1 \partial x_p} \\ \vdots & & \vdots \\ \frac{\partial^2 a(x,y)}{\partial y_r \partial x_1} & \cdots & \frac{\partial^2 a(x,y)}{\partial y_r \partial x_p} \end{bmatrix}.$$

Commonly, the Laplacian Δ_x^x is referred to as the Hessian operator, $\mathcal{H}_x$. This notation, however, will not be used herein. The Hessian of the logarithm of a Gaussian density function frequently appears in expressions for the Cramér-Rao bound. Straightforwardly, we have that

$$\Delta_x^x \log \mathrm{N}(x; \mu, P) = -P^{-1}$$

for Gaussian densities with mean μ and positive definite covariance matrix P.

4.A.2 Two Common Matrix Inversion Relations

The inverse of a 2×2 block matrix can be expressed in the inverse of its upper left block. If A^{-1} exists, the following formula holds

$$\begin{bmatrix} A & D \\ C & B \end{bmatrix}^{-1} = \begin{bmatrix} A^{-1} + A^{-1}D\Delta^{-1}CA^{-1} & -A^{-1}D\Delta^{-1} \\ -\Delta^{-1}CA^{-1} & \Delta^{-1} \end{bmatrix} \tag{4.A.19}$$

where $\Delta = B - CA^{-1}D$. The matrix Δ is known as the Schur complement of A in the block matrix $\left[\begin{smallmatrix} A & D \\ C & B \end{smallmatrix}\right]$. Another expression frequently utilized in this chapter describes the inverse of a sum of two matrices. Assuming that A^{-1} and C^{-1} exist, it holds that

$$(A + BCD)^{-1} = A^{-1} - A^{-1}B(DA^{-1}B + C^{-1})^{-1}DA^{-1} \tag{4.A.20}$$

for matrices of compatible dimension. This relation is also known as the matrix inversion lemma.

4.B Proof of General Cramér-Rao Bounds

This section summarizes the proof of the parametric and the posterior Cramér-Rao bounds given in Theorem 4.1 and Theorem 4.2, respectively. We start with a lemma concerning both theorems.

Lemma 4.B.1
Given a function $f : \mathbb{R}^{n+p} \to \mathbb{R}_+$ the following expression holds

$$\nabla_x \nabla_x^T \log f(x,y) = \frac{1}{f(x,y)} \nabla_x \nabla_x^T f(x,y) - \nabla_x \log f(x,y) \nabla_x^T \log f(x,y).$$

Proof The scalar relation $\frac{\partial}{\partial x} \log f(x) = \frac{1}{f(x)} \frac{\partial f}{\partial x}$ straightforwardly generalizes to the vector form

$$\nabla_x \log f(x,y) = \frac{1}{f(x,y)} \nabla_x f(x,y). \tag{4.B.21}$$

Inserting this relation repeatedly into the left hand side of the lemma statement yields

$$\begin{aligned}
\nabla_x \nabla_x^T \log f(x,y) &= \nabla_x \left(\frac{1}{f(x,y)} \nabla_x^T f(x,y) \right) \\
&= \frac{1}{f(x,y)} \nabla_x \nabla_x^T f(x,y) + \nabla_x \left(\frac{1}{f(x,y)} \right) f(x,y) \nabla_x^T \log f(x,y) \\
&= \frac{1}{f(x,y)} \nabla_x \nabla_x^T f(x,y) - \nabla_x \log f(x,y) \nabla_x^T \log f(x,y).
\end{aligned}$$

□

The following corollary determines an alternative expression for the FIM.

Corollary 4.B.3
Let $y \in \mathbb{R}^p$ be a random vector with density $p(y \,|\, x)$ parameterized by the non-random vector $x \in \mathbb{R}^n$, then

$$\mathsf{E}\big(\nabla_x \log p(y \,|\, x) \, \nabla_x^T \log p(y \,|\, x) \big) = \mathsf{E}(-\Delta_x^x \log p(y \,|\, x)) \,.$$

Proof Follows from Lemma 4.B.1 by taking expectation with respect to y on both sides since

$$\mathsf{E}\left(\frac{1}{p(y \,|\, x)} \nabla_x \nabla_x^T p(y \,|\, x) \right) = \int_{\mathbb{R}^p} \nabla_x \nabla_x^T p(y \,|\, x) \, dy = \nabla_x \nabla_x^T \int_{\mathbb{R}^p} p(y \,|\, x) \, dy = 0.$$

□

A similar corollary holds for the random case.

Corollary 4.B.4
Let $x \in \mathbb{R}^n$, $y \in \mathbb{R}^p$ be random vectors with joint density $p(x,y)$, then

$$\mathsf{E}\big(\nabla_x \log p(x,y) \, \nabla_x^T \log p(x,y) \big) = \mathsf{E}(-\Delta_x^x \log p(x,y)) \,.$$

Proof Follows from Lemma 4.B.1 since

$$\mathsf{E}\left(\frac{1}{p(x,y)} \nabla_x \nabla_x^T p(x,y) \right) = \int_{\mathbb{R}^{n+p}} \nabla_x \nabla_x^T p(x,y) \, dxdy = \nabla_x \nabla_x^T \int_{\mathbb{R}^{n+p}} p(x,y) \, dxdy = 0.$$

□

4.B.1 Proof of Theorem 4.1

A proof of the parametric Cramér-Rao bound follows. Inserting the coordinate transformation $x = t(z)$ into the expression for the unbiasedness of the estimator yields that

$$\int_{\mathbb{R}^p} (\hat{x}(y) - x)^T p(y \,|\, x)\, dy = \int_{\mathbb{R}^p} (\hat{x}(y) - t(z))^T p(y \,|\, z)\, dy = 0.$$

Taking the gradient w.r.t. z on both sides of the last relation above yields that

$$\int_{\mathbb{R}^p} \nabla_z \left((\hat{x}(y) - t(z))^T p(y \,|\, z) \right)\, dy = 0$$

$$\int_{\mathbb{R}^p} \nabla_z \left(p(y \,|\, z) \right) (\hat{x}(y) - t(z))^T\, dy - \int_{\mathbb{R}^p} \nabla_z t^T(z) p(y \,|\, z)\, dy = 0.$$

The cross correlation of the stochastic score vector and the estimation error thus becomes

$$\int_{\mathbb{R}^p} \nabla_z \left(\log p(y \,|\, z) \right) (\hat{x}(y) - t(z))^T p(y \,|\, z)\, dy = \nabla_z t^T(z) \tag{4.B.22}$$

after applying (4.B.21). Consider the random vector

$$\begin{bmatrix} \hat{x}(y) - t(z) \\ \nabla_z \log p(y \,|\, z) \end{bmatrix}$$

as a function of the observations y. Since this vector has zero mean, its covariance matrix is

$$\mathsf{E}\left(\begin{bmatrix} \hat{x} - x \\ \nabla_z \log p(y \,|\, z) \end{bmatrix} \begin{bmatrix} \hat{x} - x \\ \nabla_z \log p(y \,|\, z) \end{bmatrix}^T \right) = \begin{bmatrix} C & M \\ M^T & J \end{bmatrix} \tag{4.B.23}$$

where the expectation is performed over y using the likelihood $p(y \,|\, z)$. Inserting $z^\star$ into (4.B.23), we can identify the error covariance matrix

$$C = \mathsf{E}\big((\hat{x} - x^\star)(\hat{x} - x^\star)^T \big),$$

the Fisher information matrix

$$J = \mathsf{E}(-\Delta_z^z \log p(y \,|\, z))\Big|_{z=z^\star}$$

from Corollary 4.B.3, and the matrix of the gradient of the coordinate transformation

$$M^T = \nabla_z t^T(z)\Big|_{z=z^\star}$$

from (4.B.22). The covariance matrix (4.B.23) is positive semidefinite by construction. Since this matrix can be diagonalized by a nonsingular coordinate transformation

$$\begin{bmatrix} I & -MJ^{-1} \\ 0 & I \end{bmatrix} \begin{bmatrix} C & M \\ M^T & J \end{bmatrix} \begin{bmatrix} I & 0 \\ -J^{-1}M^T & I \end{bmatrix} = \begin{bmatrix} C - MJ^{-1}M^T & 0 \\ 0 & J \end{bmatrix}$$

we have that

$$C \geq MJ^{-1}M^T.$$

This can also be found directly from the Schur complement of (4.B.23). □

4.B.2 Proof of Theorem 4.2

Proof of the posterior Cramér-Rao bound follows. The estimator bias is defined by

$$B^T(z) = \int_{\mathbb{R}^p} (\hat{x}(y) - t(z))^T p(y \,|\, z)\, dy.$$

Multiplying both sides of this expression by $p(z)$ and taking the gradient w.r.t. z yields the matrix relation

$$\begin{aligned} \nabla_z \left(B^T(z) p(z) \right) &= \nabla_z \int_{\mathbb{R}^p} (\hat{x}(y) - t(z))^T p(z, y)\, dy \\ &= -\nabla_z t^T(z) p(z) + \int_{\mathbb{R}^p} \nabla_z \left(p(z, y) \right) (\hat{x}(y) - t(z))^T dy \end{aligned}$$

Integrating both sides w.r.t. z over its complete range $\mathbb{R}^r$ yields

$$\begin{aligned} \int_{\mathbb{R}^r} \nabla_z \left(B^T(z) p(z) \right)\, dz = \int_{\mathbb{R}^r} & -\nabla_z t^T(z) p(z)\, dz \\ & + \int_{\mathbb{R}^{p+r}} \nabla_z \left(p(z, y) \right) (\hat{x}(y) - t(z))^T dy dz \end{aligned} \quad (4.B.24)$$

The (i, j)-element of the left hand side matrix is

$$\int_{\mathbb{R}^r} \frac{\partial B_j(z) p(z)}{\partial z_i}\, dz = \int_{\mathbb{R}^{r-1}} \left(B_j(z) p(z) \Big|_{z_i = \infty} - B_j(z) p(z) \Big|_{z_i = -\infty} \right) dz_{\neg i}$$

where $dz_{\neg i} = dz_1 \ldots dz_{i-1} dz_{i+1} \ldots dz_r$. Under the conditions on the bias given in the theorem all these entries are zero. Finally, using (4.B.21) on the right hand side of (4.B.24), we have

$$\int_{\mathbb{R}^{p+r}} \nabla_z \log p(z, y) (\hat{x}(y) - t(z))^T p(z, y)\, dz dy = \int_{\mathbb{R}^r} \nabla_z t^T(z) p(z)\, dz, \quad (4.B.25)$$

which is the cross correlation between the score function and the estimator error. Similarly to the parametric case, consider the random vector

$$\begin{bmatrix} \hat{x} - t(z) \\ \nabla_z \log p(z, y) \end{bmatrix}$$

which is a function of the two random vectors z and y. The error correlation matrix

$$\mathsf{E}\left(\begin{bmatrix} \hat{x} - x \\ \nabla_z \log p(z, y) \end{bmatrix} \begin{bmatrix} \hat{x} - x \\ \nabla_z \log p(z, y) \end{bmatrix}^T \right) = \begin{bmatrix} C & M \\ M^T & J \end{bmatrix} \tag{4.B.26}$$

is positive semidefinite by construction. Equation (4.B.25) gives that

$$M^T = \mathsf{E}\big(\nabla_z t^T(z)\big)$$

while Corollary 4.B.4 yields that

$$J = \mathsf{E}(-\Delta_z^z \log p(z, y)) \,.$$

The result follows by a similar argumentation as in the parametric case, presented in Appendix 4.B.1.

4.C Proof of Bounds for Recursive Estimation

4.C.1 Proof of Theorem 4.3

The likelihood for the one step ahead prediction of the complete state history is determined by the recursive expression

$$p(\mathbb{Y}_t, \mathbb{Z}_t \,|\, \mathbb{U}_t) = p(y_t \,|\, \mathbb{U}_{t-1}) p(z_t \,|\, u_t) p(\mathbb{Y}_{t-1}, \mathbb{Z}_{t-1} \,|\, \mathbb{U}_{t-1}) \tag{4.C.27}$$

where the stacked vectors are defined in (4.6) and (4.9), and the decomposition follows from the Markovian property of the model (4.4). The conditions given on the model (4.4) ensure that the likelihood $p(y_t \,|\, x_t)$ (4.5) is a well defined probability density function. Therefore, the first factor of (4.C.27) is alternatively given as

$$p(y_t \,|\, \mathbb{U}_{t-1}) = p_{y_t|x_t}(y_t \,|\, \varphi_{t-1}(\mathbb{U}_{t-1}))$$

where we utilize the mapping defined in (4.7). A decomposition of the corresponding FIM with respect to the historical and predictive part of the parameters yields a recursive expression for this matrix,

$$\begin{aligned} J_t(\mathbb{U}_t) &= \mathsf{E}\Big(-\Delta_{\mathbb{U}_t}^{\mathbb{U}_t} \log p(\mathbb{Y}_t, \mathbb{Z}_t \,|\, \mathbb{U}_t) \Big) \\ &= \mathsf{E}\left(- \begin{bmatrix} \Delta_{\mathbb{U}_{t-1}}^{\mathbb{U}_{t-1}} & \Delta_{\mathbb{U}_{t-1}}^{u_t} \\ \Delta_{u_t}^{\mathbb{U}_{t-1}} & \Delta_{u_t}^{u_t} \end{bmatrix} \log p(\mathbb{Y}_t, \mathbb{Z}_t \,|\, \mathbb{U}_t) \right) \end{aligned}$$

$$= \begin{bmatrix} W_{t-1}^T(\mathbb{U}_{t-1}) H_t^T(x_t) R_t^{-1}(x_t) H_t(x_t) W_{t-1}(\mathbb{U}_{t-1}) + J_{t-1}(\mathbb{U}_{t-1}) & 0 \\ 0 & Q_t^{-1}(w_t) \end{bmatrix}.$$

In the recursion above, zeros stand for zero matrices of appropriate dimension,

$$W_{t-1}^T(\mathbb{U}_{t-1}) = \nabla_{\mathbb{U}_{t-1}} \varphi_{t-1}^T(\mathbb{U}_{t-1})$$

is a $(tm + n \times n)$ matrix that depends on $\mathbb{U}_{t-1}$, and

$$\begin{aligned} H_t^T(x_t) R_t^{-1}(x_t) H_t(x_t) &= \mathsf{E}\big(-\Delta_{x_t}^{x_t} p(y_t \,|\, x_t)\big) \\ Q_t^{-1}(w_t) &= \mathsf{E}\big(-\Delta_{w_t}^{w_t} p(z_t \,|\, w_t)\big)\,. \end{aligned}$$

The factorization above is chosen such that $R_t(x_t)$ is positive definite, and expectation is performed with respect to $p(y_t \,|\, x_t)$ and $p(z_t \,|\, w_t)$, respectively. Since $\varphi_t(\mathbb{U}_t) = f_t(\varphi_{t-1}(\mathbb{U}_{t-1}), w_t)$, the gradient of the state x_{t+1} with respect to $\mathbb{U}_t$ is recursively defined by

$$W_t^T(\mathbb{U}_t) = \begin{bmatrix} W_{t-1}^T(\mathbb{U}_{t-1}) F_t^T(\mathbb{U}_t) \\ G_t^T(\mathbb{U}_t) \end{bmatrix}$$

where

$$F_t^T(\mathbb{U}_t) = \nabla_{x_t} f_t^T(x_t, w_t) \qquad \text{and} \qquad G_t^T(\mathbb{U}_t) = \nabla_{w_t} f_t^T(x_t, w_t).$$

Straightforwardly, this yields a recursion

$$M_t^T(\mathbb{U}_t) = \nabla_{\mathbb{U}_t} \Phi_t^T(\mathbb{U}_t) = \begin{bmatrix} M_{t-1}^T(\mathbb{U}_{t-1}) & W_{t-1}^T(\mathbb{U}_{t-1}) F_t^T(\mathbb{U}_t) \\ 0 & G_t^T(\mathbb{U}_t) \end{bmatrix}. \tag{4.C.28}$$

The true state trajectory is given either as $\mathbb{U}_t^\star$ or $\mathbb{X}_{t+1}^\star$. After inserting this trajectory into the expressions above, we drop the arguments of these matrices. The Cramér-Rao bound for the estimation of the complete state sequence given in (4.11) and (4.12) becomes

$$\begin{aligned} \mathbb{P}_{t+1} &= M_t J_t^{-1} M_t^T \\ &= \begin{bmatrix} M_{t-1} & 0 \\ F_t W_{t-1} & G_t \end{bmatrix} \begin{bmatrix} W_{t-1}^T H_t^T R_t^{-1} H_t W_{t-1} + J_{t-1} & 0 \\ 0 & Q_t^{-1} \end{bmatrix}^{-1} \begin{bmatrix} M_{t-1}^T & W_{t-1}^T F_t^T \\ 0 & G_t^T \end{bmatrix}. \end{aligned}$$

The lower right sub-matrix P_{t+1} is found correspondingly to (4.13) as

$$\begin{aligned} P_{t+1} &= F_t W_{t-1} \left(W_{t-1}^T H_t^T R_t^{-1} H_t W_{t-1} + J_{t-1} \right)^{-1} W_{t-1}^T F_t^T + G_t Q_t G_t^T \\ &= F_t \left(P_t - P_t H_t^T (H_t P_t H_t^T + R_t)^{-1} H_t P_t \right) F_t^T + G_t Q_t G_t^T \end{aligned}$$

since $W_{t-1} J_{t-1}^{-1} W_{t-1}^T = P_t$.

4.C.2 Proof of Theorem 4.5

Since $p(x_{t+1} \,|\, x_t)$ is assumed to be nonsingular, the prior density $p(\mathbb{X}_{t+1})$ is well defined and there is no need to phrase the Cramér-Rao bound in terms of $\mathbb{U}_t$. From (4.4) and (4.6), we have that the joint density is recursively defined by

$$p(\mathbb{Y}_t, \mathbb{X}_{t+1}) = p(y_t \,|\, x_t) p(x_{t+1} \,|\, x_t) p(\mathbb{Y}_{t-1} \,|\, \mathbb{X}_t) \tag{4.C.29}$$

initiated by $p(\mathbb{Y}_{-1} \,|\, \mathbb{X}_0) = p(x_0)$. The Cramér-Rao bound now simply becomes the inverse of the "information matrix"

$$J_t = \mathsf{E}\left(- \begin{bmatrix} \Delta^{\mathbb{X}_{t-1}}_{\mathbb{X}_{t-1}} & \Delta^{x_t}_{\mathbb{X}_{t-1}} \\ \Delta^{\mathbb{X}_{t-1}}_{x_t} & \Delta^{x_t}_{x_t} \end{bmatrix} \log p(\mathbb{Y}_{t-1}, \mathbb{X}_t) \right) = \begin{bmatrix} A_t & B_t \\ B_t^T & C_t \end{bmatrix}. \tag{4.C.30}$$

Inserting (4.C.29) yields the recursive update

$$\begin{aligned} J_{t+1} &= \mathsf{E}\left(- \begin{bmatrix} \Delta^{\mathbb{X}_{t-1}}_{\mathbb{X}_{t-1}} & \Delta^{x_t}_{\mathbb{X}_{t-1}} & \Delta^{x_{t+1}}_{\mathbb{X}_{t-1}} \\ \Delta^{\mathbb{X}_{t-1}}_{x_t} & \Delta^{x_t}_{x_t} & \Delta^{x_{t+1}}_{x_t} \\ \Delta^{\mathbb{X}_{t-1}}_{x_{t+1}} & \Delta^{x_t}_{x_{t+1}} & \Delta^{x_{t+1}}_{x_{t+1}} \end{bmatrix} \log p(\mathbb{Y}_{t-1}, \mathbb{X}_t) \right) \\ &= \begin{bmatrix} A_t & B_t & 0 \\ B_t^T & C_t + R_t + V_t & S_t \\ 0 & S_t^T & Q_t \end{bmatrix}, \end{aligned} \tag{4.C.31}$$

where zeros indicate block matrices with zero entries of appropriate dimension, and

$$\begin{aligned} V_t &= \mathsf{E}\big(-\Delta^{x_t}_{x_t} \log p(x_{t+1} \,|\, x_t)\big) & R_t &= \mathsf{E}\big(-\Delta^{x_t}_{x_t} \log p(y_t \,|\, x_t)\big) \\ S_t &= \mathsf{E}\big(-\Delta^{x_{t+1}}_{x_t} \log p(x_{t+1} \,|\, x_t)\big) & Q_t &= \mathsf{E}\Big(-\Delta^{x_{t+1}}_{x_{t+1}} \log p(x_{t+1} \,|\, x_t)\Big). \end{aligned}$$

The Cramér-Rao bound for the one step ahead prediction problem is found in the lower right corner of the information matrix,

$$J_t^{-1} = \begin{bmatrix} * & * \\ * & P_t \end{bmatrix}.$$

Equation (4.C.30) yields that $P_t^{-1} = C_t - B_t^T A_t^{-1} B_t$, and thus we have

$$\begin{aligned} P_{t+1}^{-1} &= Q_t - \begin{bmatrix} 0 & S_t^T \end{bmatrix} \begin{bmatrix} A_t & B_t \\ B_t^T & C_t + R_t + V_t \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ S_t \end{bmatrix} \\ &= Q_t - S_t^T \left(P_t^{-1} + R_t + V_t \right)^{-1} S_t \end{aligned}$$

which concludes the proof.

4.C.3 Proof of Theorem 4.6

The joint density is given by

$$p(\mathbb{Y}_t, \mathbb{U}_t) = p(y_t \,|\, \mathbb{U}_{t-1}) p(w_t) p(\mathbb{Y}_{t-1} \,|\, \mathbb{U}_{t-1})$$

and the "information matrix" therefore given as

$$J_t = \mathsf{E}\Big(-\Delta^{\mathbb{U}_t}_{\mathbb{U}_t} \log p(\mathbb{Y}_t, \mathbb{U}_t)\Big) = \mathsf{E}\left(-\begin{bmatrix} \Delta^{\mathbb{U}_{t-1}}_{\mathbb{U}_{t-1}} & \Delta^{u_t}_{\mathbb{U}_{t-1}} \\ \Delta^{\mathbb{U}_{t-1}}_{u_t} & \Delta^{u_t}_{u_t} \end{bmatrix} \log p(\mathbb{Y}_t, \mathbb{U}_t)\right)$$
$$= \begin{bmatrix} W_{t-1} Z_t W_{t-1}^T + J_{t-1} & 0 \\ 0 & Q_t^{-1} \end{bmatrix} \tag{4.C.32}$$

where $W_t^T = \nabla_{\mathbb{U}_t} \varphi_{t-1}^T(\mathbb{U}_t)$ is independent of the parameters $\mathbb{U}_t$ the assumptions given in the theorem. In (4.C.32),

$$Z_t = \mathsf{E}\big(-\Delta^{x_t}_{x_t} \log p(y_t \,|\, x_t)\big) \qquad Q_t^{-1} = \mathsf{E}\big(-\Delta^{w_t}_{w_t} \log p(w_t)\big)$$

where expectation is performed with respect to both y_t and x_t, and w_t, respectively. The gradient matrix $M_t^T = \nabla_{\mathbb{U}_t} \Phi_t^T(\mathbb{U}_t)$ is also constant and recursively defined by

$$M_t^T = \begin{bmatrix} M_{t-1}^T & W_{t-1} F_t^T \\ 0 & G_t^T \end{bmatrix} \tag{4.C.33}$$

where F_t and G_t originate from the linear state transition relation. Analogously to the derivation in Appendix 4.C.1 for the parametric case, the Cramér-Rao bound matrix for the estimation of the complete state sequence $\mathbb{X}_{t+1}$ is given by $\mathbb{P}_{t+1} = M_t J_t^{-1} M_t^T$, and recursively determined as

$$\mathbb{P}_{t+1} = \begin{bmatrix} M_{t-1} & 0 \\ F_t W_{t-1} & G_t \end{bmatrix} \begin{bmatrix} W_{t-1} Z_t W_{t-1}^T + J_{t-1} & 0 \\ 0 & Q_t^{-1} \end{bmatrix}^{-1} \begin{bmatrix} M_{t-1}^T & W_{t-1} F_t^T \\ 0 & G_t^T \end{bmatrix}$$

after inserting (4.C.33) and (4.C.32). The lower right sub-matrix of $\mathbb{P}_{t+1}$ is the sought Cramér-Rao bound P_{t+1}, which becomes

$$\begin{aligned} P_{t+1} &= F_t W_{t-1} \left(W_{t-1}^T Z_t W_{t-1} + J_{t-1}\right)^{-1} W_{t-1}^T F_t^T + G_t Q_t G_t^T \\ &= F_t \left(P_t - P_t (P_t + Z_t^{-1})^{-1} P_t\right) F_t^T + G_t Q_t G_t^T \end{aligned}$$

since $W_{t-1} J_{t-1}^{-1} W_{t-1}^T = P_t$.

4.C.4 Proof of Theorem 4.7

The forward recursion P_t follows from Theorem 4.5 as the lower right block of the inverse of the sequence of information matrices, J_{t+1}. This information matrix has a strong diagonal structure imposed by the recursive update (4.C.31). In the

notation of Appendix 4.C.2, for any $t \leq N$ the matrix can be written

$$J_{N+1} = \left[\begin{array}{cc|ccccc} A_t & B_t & 0 & 0 & 0 & 0 \\ B_t^T & C_t + R_t + V_t & S_t & 0 & 0 & 0 \\ \hline 0 & S_t^T & Q_t + R_{t+1} + V_{t+1} & S_{t+1} & 0 & 0 \\ 0 & 0 & S_{t+1}^T & \ddots & \ddots & 0 \\ 0 & 0 & 0 & \ddots & \ddots & S_N \\ 0 & 0 & 0 & 0 & S_N^T & Q_N \end{array}\right] = \left[\begin{array}{c|c} J_{t|t} & K_t \\ \hline K_t^T & L_{t|N} \end{array}\right] \tag{4.C.34}$$

and its inverse is similarly decomposed as

$$J_{N+1}^{-1} = \left[\begin{array}{ccc|ccc} P_{0|N} & & & & & \\ & \ddots & & & & \\ & & P_{t|N} & & & \\ \hline & & & P_{t+1|N} & & \\ & & & & \ddots & \\ & & & & & P_{N+1|N} \end{array}\right] \tag{4.C.35}$$

where all unimportant entries have been left empty. It is the diagonal blocks indicated in (4.C.35) that constitute the Cramér-Rao bound for the smoothing problem. Introduce the notation $P_{t|t}$ for the lower right block of $J_{t|t}^{-1}$,

$$J_{t|t}^{-1} = \begin{bmatrix} A_t & B_t \\ B_t^T & C_t + R_t + V_t \end{bmatrix}^{-1} = \begin{bmatrix} * & * \\ * & P_{t|t} \end{bmatrix}. \tag{4.C.36}$$

From Appendix 4.C.2 we know that $P_t^{-1} = C_t - B_t^T A_t^{-1} B_t$, and therefore that

$$P_{t|t}^{-1} = R_t + V_t + C_t - B_t^T A_t^{-1} B_t = R_t + V_t + P_t^{-1}.$$

Assume that the matrices $P_{i|N}$ for $i > t$ have been computed, the next matrix in the diagonal of (4.C.35) is $P_{t|N}$. Applying the block matrix inversion formula (4.A.19) to (4.C.34), the matrix $P_{t|N}$ in (4.C.35) is found as

$$\begin{aligned} P_{t|N} &= \begin{bmatrix} 0 & I \end{bmatrix} \left(J_{t|t}^{-1} + J_{t|t}^{-1} K_t \begin{bmatrix} P_{t+1|N} & & \\ & \ddots & \\ & & P_{N+1|N} \end{bmatrix} K_t^T J_{t|t}^{-1} \right) \begin{bmatrix} 0 \\ I \end{bmatrix} \\ &= P_{t|t} + P_{t|t} S_t P_{t+1|N} S_t^T P_{t|t} \end{aligned}$$

where the last step follows from (4.C.36) and the fact that K_t, defined in (4.C.34), only will pick out $P_{t+1|N}$ for the resulting recursion. The proof follows by induction since, by definition, $P_{N+1|N} = P_{N+1}$ from the forward recursion.

5

Grid Based Methods

Statistical inference under the Bayesian paradigm relies heavily on the ability to evaluate integrals over the parameter space. For generic inference problems, these integrals typically determine the normalizing factor of the posterior density and the minimum mean-square error estimate. In recursive estimation, an additional integral implicitly defines the posterior filtering density. Since the integrals encountered in Bayesian estimation seldom have analytical solutions, practical Bayesian inference is generally an issue of approximative evaluation of integrals.

Approximative integration is usually performed using a "quadrature formula," which is a linear combination of values of the integrand. In one dimensional integration a generic quadrature is given by

$$\int_a^b f(x)\,dx \approx w_1 f(x_1) + \cdots + w_n f(x_n),$$

where, occasionally, derivatives of the integrand occur in the quadrature. The aim of the numerical integration procedure is to pick as few points x_i as possible to minimize the computational requirements and to choose the corresponding weights w_i to minimize the error in the right hand side estimate of the integral. The quadrature above is scalar; extension to higher dimensional integral approximations is straightforward but the choice of the quadrature points, x_i, and weights, w_i, becomes severely more complicated. Moreover, the number of integrand evaluations will in general increase drastically with the dimension of the integration region,

given a fixed relative error. Some brief comments on the latter fact are presented in Section 5.1.

For numerical integration on the real line, or in the plane, there exists a variety of methods to construct efficient quadratures for general integrands. These methods determine the quadrature points adaptively given the integrand and integration region. Some of these methods are also available in commercial numerical integration software. Generally, numerical integration in low dimension is strongly connected to interpolation of functions. The locations of the interpolation points are determined by the local oscillations of the integrand and the weights by the interpolation method. Special care must be taken to integrands with singularities and rapid local behavior and to infinite integration intervals. Much effort is also spent on estimating bounds on the achieved integration error. The accuracy one can hope to achieve in higher dimensional problems will in general be rather low compared to the one obtained with scalar integrals. In high dimension, interpolation methods become less feasible and instead the method of Monte Carlo integration more attractive. Monte Carlo, or sampling, techniques are presented in detail in Chapter 6.

It is an impossible task to give general guidelines on how to perform numerical approximation of the integrals in Bayesian inference. The appropriate choice of numerical integration method will depend on the dimension and shape of the integration region, on the type of integrand, and on the required accuracy of the result. This choice will therefore inevitably depend on the specific estimation problem at hand. Any textbook on numerical integration methods can be consulted for guidelines and algorithms that might apply to the current problem of interest. Davis and Rabinowitz [45, 46] give both a theoretical foundation and practical guidelines to numerical integration methods.

In recursive estimation, one additional aspect complicates the application of standard numerical integration techniques. After numerically evaluating the integrals of the first iteration of the Bayesian recursion, the integrand used in the following iterations is only known approximately. To see this, consider a recursive estimation problem on the form given in Proposition 3.2, i.e., the model is defined by explicit analytical expressions for $p(x_0)$, $p(y_t \,|\, x_t)$, and $p(x_{t+1} \,|\, x_t)$ for all $t \geq 0$. After observing the first measurement y_0, a numerical integration procedure is applied to the Bayesian recursion of Theorem 3.3. The normalizing factor of the posterior is determined by the integral

$$\alpha = \int_{\mathbb{R}^n} p(y_0 \,|\, x_0) p(x_0) \, dx_0.$$

This integral can be numerically evaluated using any suitable method from [45, 46], since the integrand consists of analytically known functions from the model of the inference problem. The time update integral at this early stage of the recursive inference is also an integral with an analytically known expression for the integrand,

$$p(x_1 \,|\, y_0) = \int_{\mathbb{R}^n} p(x_1 \,|\, x_0) p(x_0 \,|\, y_0) \, dx_0 = \alpha^{-1} \int_{\mathbb{R}^n} p(x_1 \,|\, x_0) p(y_0 \,|\, x_0) p(x_0) \, dx_0.$$

However, this integral has no single numerical solution since it is parameterized in the state vector x_1. The integration result is the posterior density function which will be approximately updated by the numerical integration method in future iterations of the Bayesian recursion. A linear combination of local basis functions is commonly used to approximate this posterior density,

$$p(x_1 \mid y_0) \approx \sum_{i=1}^{N} w_i g_i(x_1). \tag{5.1}$$

Independent of whichever numerical integration method used, the prior at the next iteration is only known approximately. During all future iterations of the Bayesian recursion, the integrals that should be approximately evaluated will all have integrands that are not analytically known. Most results on numerical integration rely on the assumption that the integrand that can be evaluated without error at any point in the integration region. In recursive estimation, an approximation of the type (5.1) must be used instead of the actual integrand. The weights w_i, and possible scale and location parameters of the basis functions in (5.1) constitute a finite set of parameters that are updated with each iteration of the Bayesian recursion. Usually, the basis functions are chosen from a fixed or adaptively determined grid mesh over the interesting part of the state space. The grid based methods utilize numerical integration to solve the integrals of the Bayesian recursion and function approximation procedures to describe the posterior density.

The practical choice of interpolation function and numerical integration method will always be problem specific. To illustrate the techniques of grid based numerical integration, Section 5.2 contains a detailed description of a numerical integration procedure specifically developed for the terrain navigation problem. The same grid based procedure was briefly described in Chapter 2. In Section 5.3, we briefly discuss how to increase the efficiency of the grid approach by introducing spatial adaptation of the grid mesh.

5.1 The Curse of Dimensionality

The main argument against attacking Bayesian estimation problems by direct numerical approximation of the involved integrals is that numerical integration in general d-dimensional Euclidean space is computationally expensive. It can be shown that, for a given level of accuracy, the computational requirements grow exponentially with the state dimension when applying a rudimentary integration method. A justification of this effect, originally due to Davis and Rabinowitz [45], follows.

As a simple example to illustrate the complexity issues of multivariate integration we consider the problem of determining the "volume" V of a region $\mathcal{B} \subseteq \mathbb{R}^d$, where $\mathcal{B} \supset \mathbb{R}^n$ for all $n < d$. That is, we seek an approximation to the multidi-

mensional integral

$$V = \int_{\mathcal{B}} dx.$$

Divide the space $\mathbb{R}^d$ into a cubical grid of atomic regions with side h, and let N denote the number of cubes that fall inside the region $\mathcal{B}$. A primitive numerical integration algorithm could determine the number N by exhaustively counting every box inside $\mathcal{B}$ and estimating V by $\hat{V} = Nh^d$. Thus, with some abuse of notation we may write

$$V = O(Nh^d).$$

Let S denote the surface measure of the boundary of $\mathcal{B}$,

$$S = \int_{\partial\mathcal{B}} dx.$$

The error $\varepsilon = V - \hat{V}$ is approximately the volume of all boxes that pass through the boundary $\partial\mathcal{B}$,

$$\varepsilon = O(hS).$$

The value of S relative to V may be found by considering that the "average volume per dimension" is of the order $V^{1/d}$, and therefore we write

$$S = O(V^{(d-1)/d}).$$

Combining the estimates above yields that the relative error made during a brute force numerical integration by applying a grid to the space, is of the order

$$\frac{\varepsilon}{V} = O(N^{-1/d}). \tag{5.2}$$

From this relation it is obvious that in order to retain the same level of relative error for numerical integrations in different dimensions the number of grid points N must increase substantially when the dimension d is increased. The exponential relation between N and d in (5.2) is usually referred to as *the curse of dimensionality*.

The error estimate (5.2) holds true for the rather primitive way of applying a uniform grid in the space $\mathbb{R}^d$ over the area $\mathcal{B}$. Some of the effect of (5.2) can be alleviated by adaptively choosing the grid point locations and thereby introducing more advanced quadrature formulas. However, general guidelines towards such an approach are hard to set. We return to this topic in Section 5.3.

In conclusion, the relative error for any numerical integration method using a uniform mesh to approximate the integration behaves as $O(N^{-1/d})$. We will show in Chapter 6 that there are alternative methods that do not suffer from this limitation. The simulation based methods of Chapter 6 have a relative error of the order $O(N^{-1/2})$, which does not explicitly depend on the state dimension. Note though that these results are asymptotic and there may very well be practical cases, especially of low dimension, where direct numerical integration is superior to simulation based integration. This fact is illuminated in Chapter 7, where an evaluation of different algorithms for terrain navigation is presented.

5.2 The Point-Mass Filter

Uniform grid approximation in recursive estimation dates back to the seminal work of Bucy and Senne [34]. The curse of dimensionality implies that these uniform mesh approximations cannot be applied in very high dimensions without adaptively choosing the grid points. Still, low dimensional problems are commonly solved using a uniform grid over the interesting part of the state space. It is mainly the ease of implementation and compact description of the grid nodes that speaks in favor of a uniform grid mesh over a non-uniform, adaptively chosen, grid. Some general guidelines on the description and construction of the grid mesh in recursive estimation are given by Simandl and Královec [133], Sorenson [137], and by Bucy and Senne [34].

Any guide to practical recursive estimation using numerical integration will be more or less problem specific. The more general class of recursive estimation problems studied, the less precise and detailed can the given guidelines be made. The reason for this trade-off between generality and explicitness is that the most efficient algorithms need to explore every detail in the structure of the problem. Most of the suggested algorithms found in the literature have been developed with a specific recursive estimation problem in mind. The proposed methods are still very similar and mainly differ only in the way that the grid is described and updated. Instead of giving a review over many different but similar grid-based algorithms for recursive estimation, we choose to present a detailed description of a numerical integration method tailor-made for the terrain navigation application. The algorithm applies a uniformly adaptive grid to the part of the state space where the filter density has its main support. Elementary masses placed in this grid nodes gives an approximative description of the posterior filter density. The method was briefly described in Chapter 2 under the name Point-Mass Filter (PMF). The grid is automatically refined and updated with each new measurement. The aim has been to design an update with a minimum of design parameters that in a natural way trade off accuracy against computational complexity.

5.2.1 Background

The concept of terrain navigation was described in some detail in Chapter 2. This navigation principle demands the recursive solution of a nonlinear estimation problem. With x_t denoting the sought aircraft position in the terrain map the underlying state space estimation model is given by (2.4),

$$\begin{aligned} x_{t+1} &= x_t + u_t + v_t \\ y_t &= h(x_t) + e_t \end{aligned} \qquad t = 0, 1 \dots \tag{5.3}$$

Since the PMF algorithm is tailor-made for the terrain navigation application, the grid update and propagation have been biased towards a grid solution well suited for this problem. This involves, e.g., the possibility to handle multi-modal filtering distributions. There are only some minor additional restrictions introduced by the PMF implementation itself. It can in general be straightforwardly applied to any

recursive estimation problem that is given by a model equivalent to (5.3). With minor adjustments, all problems with linear state transition and possibly nonlinear measurement relation can be handled with the technique of this section. The noises acting on the model may be non-Gaussian but are required to be white and independent. However, due to the implementational complexity and the curse of dimensionality it is not advisable to apply the PMF to high dimensional problems even if they fit this model structure.

The Bayesian solution to (5.3) consists of a recursive propagation of the density function for the aircraft position given the terrain elevation measurements. An estimate of the aircraft position and its corresponding error covariance is calculated from this density. These quantities are sent to a central Kalman filter that fuses navigation information from several different sources. The role of this Kalman filter is to track the internal integration errors in the Inertial Navigation System (INS) and to check the overall integrity of the navigation system by detecting malfunctions of the separate navigation sensors. The estimate from the terrain navigation filter is used as a component in the measurement vector of this central Kalman filter, and the estimation error covariance is used as the corresponding measurement error covariance. The recursive expressions for the posterior density are therefore augmented with expressions for the calculation of an estimate and its estimation error covariance. In summary, the expressions that should be computed at each iteration of the Bayesian solution to terrain navigation are given by (2.6)–(2.8),

$$\alpha_t = \int_{\mathbb{R}^2} p_{e_t}(y_t - h(x_t))\, p(x_t \,|\, \mathbb{Y}_{t-1})\; dx_t \tag{5.4a}$$

$$p(x_t \,|\, \mathbb{Y}_t) = \alpha_t^{-1} p_{e_t}(y_t - h(x_t))\, p(x_t \,|\, \mathbb{Y}_{t-1}) \tag{5.4b}$$

$$\hat{x}_t^{\mathrm{MS}} = \int_{\mathbb{R}^2} x_t p(x_t \,|\, \mathbb{Y}_t)\; dx_t \tag{5.4c}$$

$$C_t = \int_{\mathbb{R}^2} (x_t - \hat{x}_t^{\mathrm{MS}})(x_t - \hat{x}_t^{\mathrm{MS}})^T p(x_t \,|\, \mathbb{Y}_t)\; dx_t \tag{5.4d}$$

$$p(x_{t+1} \,|\, \mathbb{Y}_t) = \int_{\mathbb{R}^2} p_{v_t}(x_{t+1} - u_t - x_t)\, p(x_t \,|\, \mathbb{Y}_t)\; dx_t. \tag{5.4e}$$

The nonlinear recursive inference is performed as described by the update of the conditional density using (5.4b) and (5.4e). These expressions need to be evaluated upon reception of every new measurement y_t. The expressions (5.4a), (5.4c) and (5.4d) only have to be calculated when an estimate of the position is needed in the central Kalman filter. In the sequel it will be assumed that the estimate calculations are done at the same rate as the conditional density is updated, even though this is not a necessity.

The total algorithm of (5.4) will consist of solving four generalized integrals over $\mathbb{R}^2$. One obvious way to reduce the number of integrals in each iteration is to choose the maximum a posteriori (MAP) estimate instead of the MMSE estimate. For the computation of the MAP estimate the normalization integral (5.4a) is redundant,

and the estimate is recursively found from the unnormalized density

$$p(x_t \mid \mathbb{Y}_t) = p_{e_t}(y_t - h(x_t))\, p(x_t \mid \mathbb{Y}_{t-1}) \tag{5.5a}$$

$$\hat{x}_t^{\text{MAP}} = \arg\max_{x_t} p(x_t \mid \mathbb{Y}_t) \tag{5.5b}$$

$$p(x_{t+1} \mid \mathbb{Y}_t) = \int_{\mathbb{R}^2} p_{v_t}(x_{t+1} - u_t - x_t)\, p(x_t \mid \mathbb{Y}_t)\; dx_t. \tag{5.5c}$$

However, normalization is still needed for the computation of the estimation error covariance. Focus will here be on the MMSE estimate, i.e., a numerical a grid based numerical integration approximation to the recursion (5.4).

5.2.2 Point-Mass Approximation

There are a number of approximation schemes that will turn the Bayesian functional propagation in (5.4) into a point-mass equivalent. The common unifying factor of these schemes is that the propagation of the continuous probability density function $p(x_t \mid \mathbb{Y}_t)$ is replaced by a propagation of the probability in a finite set of grid points spread over the region of interest in the state space. Some approaches to the approximation that will end up as point-mass algorithms are:

- Discretization of the state space. Limit the range of values of x_t from the continuum $\mathbb{R}^2$ to a finite number of levels.
- Numerical approximation of the integrals. Replace the infinite integrals with Riemann sums over finite intervals.
- Probability region equivalent. Divide the state space into regions and express the probability of being in each region. Use this probability as a weight on each region.
- Piecewise constant approximation of the posterior. Numerically approximate the posterior as a sum of weighted and shifted indicator functions.
- Nyquist approach. Assume that the posterior is band-limited, i.e., has an upper bound on the frequency of spatial variation. Sample the function spatially and update these samples.

Whatever approach used to find the approximate solution, almost identical expressions for the point-mass algorithm are obtained. The integrals must naturally turn into sums over the grid point positions and specially, the convolution time update (5.4e) will turn into a discrete convolution.

In the PMF we utilize a simple numerical integration method with quadrature points given in a grid mesh of uniform resolution. The posterior density is approximately described by point-mass values located in the nodes of this grid. Assume that N grid points in $\mathbb{R}^2$ have been chosen for the approximation of $p(x_t \mid \mathbb{Y}_t)$. Introduce the notation

$$x_t(k) \qquad k = 1, 2, \ldots, N$$

for these N vectors in $\mathbb{R}^2$. Each of these N grid points are equipped with a corresponding probability mass weight

$$p(x_t(k) \,|\, \mathbb{Y}_t) \qquad k = 1, 2, \ldots, N.$$

The grid points are chosen from a uniform mesh over $\mathbb{R}^2$. The mesh has resolution δ and an outer boundary of the grid point support defined by an $(n \times m)$ matrix sparsely occupied by grid point values. Figure 5.1 exemplifies the structure of this grid point approximation. The lower left corner of the outer boundary of the

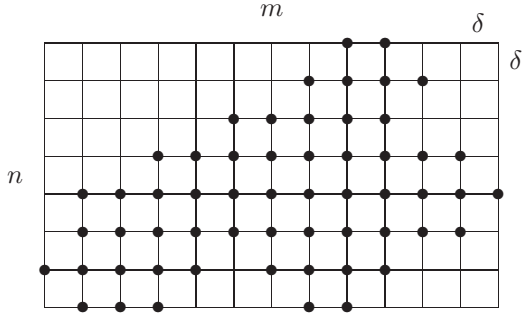


Figure 5.1: Example constellation of the two dimensional grid. The N grid points • are arranged in a mesh of size $(n \times m)$ with resolution δ.

mesh is used as a reference point $\bar{x}_t \in \mathbb{R}^2$ that defines the absolute grid mesh location in the state space. The reference vector, the scalar resolution factor and a matrix with N nonzero positive values uniquely define a data structure for the grid approximation.

The position of an arbitrary grid point $x_t(k)$ in the mesh can be found from its relative coordinates in the mesh, the grid resolution, and the reference vector. The grid mesh defines an approximation of the conditional density given by a point-mass value in each grid node position. This description induces an approximation of each integral in (5.4) by a finite sum over nonzero grid point values

$$\int_{\mathbb{R}^2} f(x_t)\, dx_t \approx \sum_{k=1}^{N} f(x_t(k))\, \delta^2.$$

Inserting this expression into (5.4) yields the Bayesian point-mass recursion:

$$\alpha_t = \sum_{n=1}^{N} p_{e_t}(y_t - h(x_t(n)))\, p(x_t(n) \,|\, \mathbb{Y}_{t-1})\ \delta^2 \tag{5.6a}$$

$$p(x_t(k) \,|\, \mathbb{Y}_t) = \alpha_t^{-1} p_{e_t}(y_t - h(x_t(k)))\, p(x_t(k) \,|\, \mathbb{Y}_{t-1}) \tag{5.6b}$$

$$\hat{x}_t^{\mathrm{MS}} = \sum_{n=1}^{N} x_t(n) p(x_t(n) \,|\, \mathbb{Y}_t)\ \delta^2 \tag{5.6c}$$

$$C_t = \sum_{n=1}^{N} (x_t(n) - \hat{x}_t^{\mathrm{MS}})(x_t(n) - \hat{x}_t^{\mathrm{MS}})^T p(x_t(n) \,|\, \mathbb{Y}_t)\ \delta^2 \tag{5.6d}$$

$$x_{t+1}(k) = x_t(k) + u_t \quad k = 1, 2, \ldots, N \tag{5.6e}$$

$$p(x_{t+1}(k) \,|\, \mathbb{Y}_t) = \sum_{n=1}^{N} p_{v_t}(x_{t+1}(k) - x_t(n))\, p(x_t(n) \,|\, \mathbb{Y}_t)\ \delta^2. \tag{5.6f}$$

Due to the simple state evolution in terrain navigation, the time update has been split into two parts. First the grid points are updated with the movement of the aircraft u_t in (5.6e), and then the filter density is convolved with the density for $\boldsymbol{v}_t$, in (5.6f). The update (5.6e) is effectively performed by adding u_t to the reference vector $\bar{x}_t$ of the approximation mesh in Figure 5.1.

As new measurements are processed, the conditional density will concentrate in an area of the aircraft position. For an efficient grid implementation, the grid mesh should also adapt its support and resolution in order to meet this continuous evolution of the conditional density. In general, the grid adaption must be considerably more elaborate than (5.6e).

5.2.3 Grid Adaption

The measurement update will decrease the value of the conditional density in areas with low likelihood while the time update will smooth the conditional density and increase its support. The grid should be adapted accordingly, removing values with low probability from the grid and increase the grid support through the time update. The resolution should be increased when enough grid points have been removed from the mesh and when measurements with low information content are received, the grid resolution should be decreased again so that the computational requirements stay inside som predefined limits. This refinement and adjustment of the grid is of great importance for an efficient algorithm implementation with small approximation errors.

Approximation errors in the numerical implementation demand a frequent normalization of the point-mass weights. Since the posterior density should integrate to unity, a consistent point-mass approximation satisfies

$$\int_{\mathbb{R}^2} p(x_t \,|\, \mathbb{Y}_t) = \sum_{k=1}^{N} p(x_t(k) \,|\, \mathbb{Y}_t)\, \delta^2 = 1. \tag{5.7}$$

The average value of a set of point-mass weights satisfying the normalization constraint depends on the number of grid points in the mesh and the mesh resolution,

$$\frac{1}{N} \sum_{k=1}^{N} p(x_t(k) \,|\, \mathbb{Y}_t) = \frac{1}{N\delta^2}.$$

A truncation parameter $\varepsilon > 0$ is introduced to control the reduction of points with low probability from the mesh. Every grid point with a weight less than ε times

the average mass value is removed from the grid after each measurement update. The new set of grid points is defined by

$$\left\{x_t(k) : p(x_t(k) \,|\, \mathbb{Y}_t) > \varepsilon/N\delta^2\right\}.$$

This can also be interpreted as removing the grid points that only contribute to the normalization sum (5.7) by some fraction ε. Let N denote the new number of grid points left in the mesh after the truncation. The weights need to be normalized after this operation in order to retain a consistent point-mass approximation,

$$p(x_t(k) \,|\, \mathbb{Y}_t) := \frac{p(x_t(k) \,|\, \mathbb{Y}_t)}{\sum_{n=1}^{N} p(x_t(n) \,|\, \mathbb{Y}_t)\, \delta^2} \qquad k = 1, \ldots, N. \tag{5.8}$$

The truncation of small mass points will make holes in the mesh, but the mesh will still have the resolution δ. If grid points at the borders of the mesh are removed the size and reference position of the mesh have to be adjusted accordingly. That is, n and m in Figure 5.1 will have to be decreased, and possible $\bar{x}_t$ adjusted.

The truncation can only decrease the number of grid points N while the time update convolution in general will increase the number N while enlarging the grid support. The time update will thus also increase the bounding mesh size, i.e., both n and m in Figure 5.1 will increase. This will be the case since the PDF $p(x_{t+1} \,|\, \mathbb{Y}_t)$ is a function of wider support than $p(x_t \,|\, \mathbb{Y}_t)$. It might happen that the measurement and time update operations will balance each other, leaving the number of grid points N from one iteration to the next almost constant. However, such a stationary state of operation will not be held for a long time since the amount of grid points that are reduced through the truncation depends on the amount of variation in the terrain. Flying from rough to flat terrain, the number of grid points will naturally increase. The opposite is true when flying from flat to rough terrain.

When the algorithm is initialized, the uncertainty about the aircraft position is high and thus the prior will have a wide support. Then it is not interesting to have a dense grid. It is more appealing to start with a sparse grid and run the algorithm until the number of remaining grid points falls below some threshold. Then the mesh resolution can be increased and the algorithm continued to process new measurements updating the PDF in the new dense grid. The denser the grid is, the better will the approximation be, which in turn will yield a higher estimation accuracy. It is also desirable, for computational load reasons, to be able to control the number of grid points used in the approximation. Let N_0 and N_1 be some lower and upper limits on the number of nonzero grid points in the mesh. These can be found from the hardware requirements put on the implementation, or seen as tuning parameters for a trade-off between algorithm performance and computational requirements. A simple control law that will try to keep the number N inside the interval $[N_0, N_1]$ is:

- If $N > N_1$ decimate the mesh, i.e., remove every second row and column in the matrix in Figure 5.1. The new resolution will be 2δ.

- If $N < N_0$ interpolate one new point-mass between every neighboring pair in the matrix in Figure 5.1. The new resolution will be $\delta/2$.

This simple law is used in the PMF for controlling the mesh resolution. The resampling of the mesh is thus only performed when the number N falls outside the user defined interval $[N_0, N_1]$. Thus, the adaption procedure of the grid in the PMF consists of two separate steps. First, grid points with low probability mass are removed from the grid, and then the grid resolution is adjusted if necessary.

5.2.4 Implementation

The point-mass values in the approximation shown in Figure 5.1 are stored in a matrix of dimension $(n \times m)$. The mesh itself is defined by a reference vector $\bar{x}_t$ for positioning the matrix in $\mathbb{R}^2$ and a grid resolution variable δ. In summary, the point-mass approximation is described by $nm + 3$ real values. However, due to the truncation operation there will be several big holes of zero probability mass inside the matrix of dimension $(n \times m)$. It would be convienient not to have to compute, e.g., the measurement update multiplication in the areas of zero probability.

Our implementation of the PMF utilizes the special purpose low level operations available for sparse matrices in the numerical package MATLAB™ [114]. Instead of storing and operating on the complete $(n \times m)$ matrix, only the nonzero elements and their indeces are stored and operated on. There is a number of overloaded low level operations available in MATLAB™, e.g., regular and element-wise matrix multiplication. Additional functions can count the number of nonzero elements in the matrix and find the indices of the nonzero elements.

All operations in (5.6) except (5.6f) are straightforward to implement using a matrix representation of the grid mesh. The measurement update (5.6b) will be an element-wise multiplication between the point-mass matrix approximation of $p(x_t \,|\, \mathbb{Y}_{t-1})$ and the a matrix containing the likelihood for y_t evaluated at the nonzero grid points of this approximation. Let $\odot$ denote element-wise, or Hadamard [90], multiplication between matrices of the same dimensions. Let $P_{t|t-1}$ be the matrix of point-masses in the approximation of $p(x_t \,|\, \mathbb{Y}_{t-1})$ and H_t be a matrix of terrain elevation values in the positions $x_t(k)$. Then the unnormalized measurement update can be written

$$P_{t|t} = p_{e_t}(y_t \mathbf{1}_{n\times m} - H_t) \odot P_{t|t-1}$$

where the evaluation of the PDF of $\boldsymbol{e}_t$ is performed element-wise on the matrix argument. The normalization (5.6a) is performed as in (5.8) by computing the sum of all entries in $P_{t|t}$. The estimate and error covariance computations in (5.6) are easily expressed as weighted sums along the columns and rows of $P_{t|t}$. The truncation of grid points with low probability is a simple search in the matrix for grid points satisfying the truncation condition.

The time update convolution (5.6f) and the interpolation scheme introduced in the adaptation of the grid are the computationally most cost-some operations of the PMF algorithm. In order to implement these operations efficiently, some additional features of the terrain navigation application need to be explored.

Convolution

The convolution is the computationally most burdensome operation in (5.6). However, the sparse matrix representation and the structure of the problem can be utilized to reduce the computational burden somewhat.

The convolution is performed between the point-mass approximation to $p(x_t \mid \mathbb{Y}_t)$ and the PDF $p(v_t)$. The system noise $\boldsymbol{v}_t$ models the drift in the INS between two measurements. There is no reason to believe that the probability of a drift in one direction is higher that in another. Thus, the PDF for the system noise is assumed to be rotational invariant. For a Gaussian distribution this means that it has a diagonal covariance matrix, and that the two-dimensional Gaussian PDF can be written as a product of two one-dimensional Gaussian PDFs. This makes it possible to perform the convolution in two steps, first along the rows of the matrix and then along the columns. Convolution kernels with this property are sometimes labeled separable, see [122] for more background on two-dimensional convolution.

The Gaussian PDF needs to be truncated to yield a finite convolution kernel. This truncation level can be chosen such that it will be far below the effect of the truncation variable ε introduced to adapt the grid support. Let the $(1 \times l)$ row vector $v = [v_1, v_2, \ldots, v_l]$ be the truncated one dimensional Gaussian distribution used as a convolution kernel. The convolution along the rows of the matrix can be written as the matrix multiplication

$$\underbrace{\overset{(n\times m)}{\begin{bmatrix} & & \\ & & \\ & & \end{bmatrix}}}_{P_{t|t}} \cdot \underbrace{\overset{(m\times m+l-1)}{\begin{bmatrix} v_1 & v_2 & \ldots & v_l & & & \\ & v_1 & v_2 & \ldots & v_l & & \\ & & \ddots & \ddots & & \ddots & \\ & & & v_1 & v_2 & \ldots & v_l \end{bmatrix}}}_{D_r} = \underbrace{\overset{(n\times m+l-1)}{\begin{bmatrix} & & \\ & & \\ & & \end{bmatrix}}}_{P_{t|t}D_r}$$

see [122] for details. Here D_r denotes the matrix above of dimension $(m \times m+l-1)$ with m repeated versions of the row vector v along the diagonal. Similarly let D_c denote an $(n + l - 1 \times n)$ matrix with n repeated versions of the column vector v^T along the diagonal. Then the two dimensional time update convolution can be written as a matrix multiplication

$$P_{t+1|t} = D_c P_{t|t} D_r,$$

where $P_{t|t}$ is the sparse matrix of point-masses in the approximation of $p(x_t \mid \mathbb{Y}_t)$ and $P_{t+1|t}$ is the point-mass matrix for $p(x_{t+1} \mid \mathbb{Y}_t)$. Utilizing the sparse functions in MATLAB™ this convolution will demand approximately $2Nl + l^2$ operations, this should be compared with nml^2 operations for the direct application of the convolution double sum. Since the dimensions n and m are much larger than l, and $N \leq mn$ by construction, this is a considerable computational gain.

Interpolation

As soon as the number of nonzero grid points falls under N_0, bilinear interpolation is used to increase the number N and the resolution of the mesh. Bilinear interpolation is obtained by first performing linear interpolation along the rows of the matrix followed by linear interpolation along the columns. The scheme is depicted in Figure 5.2. The empty circles ∘ denotes the original grid points while the filled

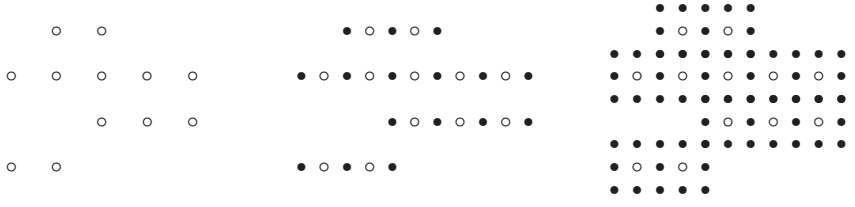


Figure 5.2: Illustration of sparse bilinear interpolation.

circles • denotes the interpolated values. If there were no holes in the grid, this interpolation would increase the number of grid points by at least a factor of four, hence the design parameters N_0 and N_1 should be more than a factor four apart.

The linearly interpolated values, • in Figure 5.2, are found by convolving the original matrix with the kernel $v = [0.5, 0.5]$. Thus, for an efficient implementation, the same algorithm as was used for the time update convolution is used for the implementation of the interpolation.

Summary of the Algorithm

The point-mass filter algorithm is initiated by an off-line computed grid mesh approximation of $p(x_0)$. The approximation of this prior density defines the initial grid mesh through the sparse matrix of point-mass values $P_{0|-1}$, the reference point $\bar{x}_0$ and the resolution parameter δ. After every approximative update of the grid points, a normalization step guarantees that the approximation is consistent with a density function integrating to unity.

Algorithm 5.1 (The Point Mass Filter (PMF))

1. Interpolate in the terrain map to find the matrix H_t of terrain elevation samples $h(x_t(k))$ with the same support as $P_{t|t-1}$.
2. Compute $P_{t|t} = p_{e_t}(y_t \mathbf{1}_{n\times m} - H_t) \odot P_{t|t-1}$. Normalize the result.
3. Calculate the estimate $\hat{x}_t^{\mathrm{MS}}$ and its error covariance C_t according to (5.6c) and (5.6d).

4. Truncate all weights that are less than $\varepsilon/N\delta^2$, where N is the number of nonzero weights in $P_{t|t}$. Normalize the remaining weights and adjust $\bar{x}_t$ if needed.
5. Determine N, the number of nonzero weights after the normalization.
   (a) If $N > N_1$, remove every second row and column in $P_{t|t}$. Set $\delta := 2\delta$. Adjust $\bar{x}_t$ if needed.
   (b) If $N < N_0$, use sparse bilinear interpolation to increase the resolution. Set $\delta := \delta/2$. Adjust $\bar{x}_t$ if needed.
6. Move the grid, $\bar{x}_{t+1} = \bar{x}_t + u_t$.
7. Build the sparse convolution matrices D_c and D_r.
8. Compute the sparse convolution $P_{t+1|t} = D_c P_{t|t} D_r$.
9. Output the estimate $\hat{x}_t^{\mathrm{MS}}$ and its error covariance C_t. Increase the time step $t := t + 1$ and continue at item 2 above.

Several features makes this algorithm well suited for use in terrain navigation and similar applications. The general description of the grid mesh allows for a posterior filter density support of very general shape. The grid can approximately describe multi-modal distributions with virtually no extra computational requirements or elaborate detections of the number of modes. The nonlinear and unstructured terrain map yields that such general distributions are very common in the terrain navigation application. The uniform grid resolution yields a simple implementation of the convolution operation which is important for real time application. In the simulations performed with our implementation, the filter runs in real time on a standard workstation. In the terrain navigation application, real time is approximately ten iterations per second. Simulations using the PMF are presented both in Chapter 2 and in Chapter 7.

5.3 Spatially Adaptive Grid

The grid resolution in the PMF is sporadically adjusted during the recursive propagation of the algorithm. At the instants of resampling, the grid mesh resolution is either divided in half, or increased to the double using a linear interpolation scheme. Thus, the resolution of the grid mesh in the PMF is adaptive with time, but uniform in space. An efficient representation of the posterior density is obtained by truncating grid points with small probability values at each algorithm recursion. The point-mass approximation is thus constrained to the uniform mesh, but is allowed to contain large areas of zero probability that do not need to be updated by the algorithm.

An even more compact description of the posterior filter density can be obtained by allowing for adaptively determined grid point locations in the state space. A grid of higher resolution would then be applied in regions where the posterior density function rapidly changes from high to low probability, or from low to high. Such

spatial adaptation would certainly be valuable in higher dimensional problems, when the curse of dimensionality comes into play.

The issue of determining a spatially adaptive grid is strongly connected to general theory of function approximation. The key is to compactly describe the filtering density, e.g., in a linear expansion such as

$$p(x_t \,|\, \mathbb{Y}_t) \approx \sum_{i=1}^{N} w_i g_i(x_t) \tag{5.9}$$

using as few terms as possible. The point-mass filter can be interpreted as a rudimentary function approximation scheme having thin indicator functions as basis elements. An adaptive grid method would correspond to using basis functions $g_i(\cdot)$ *adaptively* chosen to the function $p(x_t \,|\, \mathbb{Y}_t)$. However, we are not solely interested in minimizing N in (5.9) but also need to update this description through the Bayesian recursive solution. This puts a complicated constraint on the choice of approximation method in (5.9). Generally, this imposes that the basis functions themselves must have known analytical or approximative solutions to the propagation thorough the Bayesian recursion. From Theorem 3.3 we have the conceptual update

$$\begin{aligned} p(x_t \,|\, \mathbb{Y}_t) &\propto p(y_t \,|\, x_t) p(x_t \,|\, \mathbb{Y}_{t-1}) \\ p(x_{t+1} \,|\, \mathbb{Y}_t) &= \int_{\mathbb{R}^n} p(x_{t+1} \,|\, x_t) p(x_t \,|\, \mathbb{Y}_t)\, dx_t \end{aligned} \qquad t = 0, 1, \ldots \tag{5.10}$$

In the point-mass approach the basis functions can be regarded as infinitely thin delta measures which neither are affected by the multiplication operation nor the convolution operation in this recursive update.

Wavelet multiresolution theory [42] yields a general framework for function approximation with spatially adaptive basis functions. Moreover, wavelet techniques have also been utilized in numerical analysis for numerical solution of partial differential equations, see e.g., [27, 74, 89, 141]. The operations encountered in this application are similar to the multiplication and convolution operations in the Bayesian recursion. Thus, combining the wavelet approach to numerical solution of partial differential equations with the function approximation theory of wavelet bases would yield a general approach to adaptive grid methods in recursive Bayesian estimation. The prior density would have to be expanded in a wavelet basis, inserted into the Bayesian solution (5.10), and approximately propagated through this recursion. A wavelet basis expansion generally has the form

$$p(x_t \,|\, \mathbb{Y}_{t-1}) = \sum_k \beta_{j_0,k} \varphi_{j_0,k}(x_t) + \sum_{j=j_0}^{j_1} \sum_k \alpha_{j,k} \psi_{j,k}(x_t) \tag{5.11}$$

where $\varphi_{j,k}(\cdot)$ are *scaling functions*, while $\psi_{j,k}(\cdot)$ are the *wavelets*. The indices j and k define a dyadic grid mesh and the spatial adaptation is obtained by removing coefficients $\alpha_{j,k}$ of small magnitude, by some *wavelet shrinkage* procedure [57].

General concepts in wavelet theory are presented by Daubechies [42], Mallat [113], and Kaiser [97], while Sweldens [141] provides a presentation oriented towards wavelet applications in numerical analysis.

Assume that the prior density is given in a sparse wavelet basis, on the form given in (5.11). The measurement update step in the Bayesian recursion (5.10) consists of a multiplication with the likelihood of the observed measurement. Multiplication of functions in wavelet bases has been studied by Beylkin [26], who proposes to perform the multiplication by uncoupling the inherent connection between wavelet coefficients of different scale. The methods in [26] can be utilized for the measurement update step by first expanding the likelihood in a wavelet basis, similar to the form used for the prior (5.11).

The time update step in the recursive Bayesian solution consists of a convolution with the prior density of the process noise. This convolution time update step generally falls into a class of integral operators on the form

$$(Tf)(x) = \int K(x,y)f(y)\,dy \tag{5.12}$$

extensively studied by Beylkin et al. [28] in the case of orthogonal wavelet expansions. The operator kernel $K(x,y)$ is expressed in a wavelet basis where the location of the signification wavelet coefficients are known a priori. Both the expansion of the kernel, and the application of the operator in the wavelet basis can be performed efficiently using algorithms provided by Beylkin et al. [28].

Both Goedecker [74] and Holmström [89] emphasize the class of interpolating wavelets for numerical solution of partial differential equations. The interpolating wavelets are designed to have a strong connection between the wavelet coefficients and the interpolation error of the studied function. This connection yields some advantages when considering both local nonlinear operators, such as multiplications, and linear operators, such as (5.12). The interpolating wavelets were introduced by Deslauriers and Dubuc [54], and later reinvented and extended by Donoho [58, 59]. A practically oriented introduction to interpolating wavelets, along with a framework for design of new wavelet bases in this class, are provided by Sweldens and Schröder [142]. In [20], we present some suggestions on how to utilize interpolating wavelets in the terrain navigation application. However, the general conclusion from this work is that these methods are very complicated to implement and the overhead computations required for the adaptive grid mesh update often compensate the gains of the adaptive mesh.

In the literature utilizing wavelet techniques for the numerical solution of partial differential equations, the reported examples almost unexceptionally deal with scalar or possibly two-dimensional problems. The reason for this is not that the computational complexity grows with the dimension of the problem. Instead, it is the implementational constraints that become severe when applying the wavelet techniques to problems of high dimension. This fact is evident already in the the correspondingly low dimensional algorithms for multiplication and convolution outlined in [20], which are much more complicated than the operations in the PMF of Algorithm 5.1.

5.4 Conclusion

A numerical implementation of the Bayesian solution to recursive estimation involves numerical evaluation of integrals and integral operators. The PMF of Algorithm 5.1 as been specifically designed with the terrain navigation application in mind to yield a reasonable trade-off between an accurate density description and an efficient implementation. Each recursive estimation problem needs to be investigated in depth for the design of a corresponding point-mass filter. This will generally yield different approaches to the grid design and update rules for each application. In some cases, an adaptive grid mesh will prove to increase the algorithm performance considerably, while the gain will be less clear in other applications.

Any numerical integration method based on uniform mesh approximation will suffer from a computational penalty that increases exponentially with the state dimension. An adaptive grid mesh, e.g., introduced by wavelet shrinkage techniques, will to some extent alleviate this effect. However, the sole implementation complexity of the operations on the wavelet representations may very well overshadow the gains in effectiveness that such an approach promises.

6

Simulation Based Methods

In Chapter 3, general Bayesian estimation theory was posed as a problem of numerical integration or optimization. The standard numerical integration methods from Chapter 5 have the major limitation that the computational complexity grows exponentially with the dimension of the integration region. It was shown in Chapter 5 that the relative error of the brute force techniques is of the order $O(N^{-1/d})$, where d is the state dimension. One way to alleviate this "curse of dimensionality" was exemplified in Section 5.3. However, these adaptive methods generally lead to complicated algorithms with severe implementational difficulties. The optimization problems can of course be attacked using classical iterative techniques, like Gauss-Newton and related methods. However, the demand for good initialization of these local search methods will in general make them application specific.

Simulation based Monte Carlo integration and optimization form a suite of methods that promise general solutions to complex and high dimensional problems of numerical integration and optimization. The methods have the main advantage over classical numerical integration that the relative error is of the order $O(N^{-1/2})$, not explicitly depending on the state dimension. Still, these methods remain computer-intensive and generally put high demands on both the computational resources and the available memory size.

The simulation based methods were developed in the 1950's but it is only recently that they have been practically applied to problems of statistical inference and signal processing. In the late 1980's the introduction of cheap high performing

personal computers brought a newborn interest into numerical approximation for Bayesian inference in general, and for Monte Carlo methods in particular. This has led to a Bayesian revolution in applied statistics. Today, a large fraction of papers published in applied statistics regard topics of Monte Carlo integration and particularly Markov Chain Monte Carlo (MCMC) methods.

Monte Carlo methods have also started to appear in the signal processing and system identification literature with applications such as detection and estimation of sinusoids in noise [3, 55], and estimation of parameters in ARMA models [11, 12]. A general discussion on the applicability of Monte Carlo methods to problems in automatic control has recently been presented by Vidyasagar [153]. The field of Markov chain Monte Carlo methods is currently very active with both new theoretical and applied results appearing constantly. The latest additions on The MCMC Preprint Service [145] provide an up-to-date insight into this progressing research field.

This chapter provides a survey over the field of Monte Carlo methods in Bayesian statistics, and particularly applied to recursive estimation. The following section gives a general background to the fundamental concepts of Monte Carlo integration and optimization methods. Section 6.2 presents the classical methods of rejection and importance sampling, while Section 6.3 gives a review over Markov chain Monte Carlo methods. Finally, some Monte Carlo methods for recursive estimation are given in Section 6.4. We present Monte Carlo algorithms for terrain navigation in Chapter 7 and for target tracking in Chapter 8.

The summary of simulation based techniques for Bayesian estimation presented in this chapter is by no means a complete survey over this vast field of applied statistics. The results presented here have been gathered from several sources. The tutorial of Andrieu et al. [4] gives a detailed survey over Bayesian signal processing using simulation based techniques. Theoretical issues as well as pseudocode for several classical algorithms are presented, and examples from different applications are highlighted. A less mathematical but nevertheless utterly detailed and comprehensive review of Monte Carlo methods is given by Neal [117]. Monte Carlo inference with application to artifical intelligence and neural networks with some connections to statistical physics and a long annotated bibliography can be found in [117]. The basic technique of Monte Carlo integration, not necessarily applied to Bayesian estimation, is covered by textbooks on numerical integration such as [45, 46].

6.1 Monte Carlo Integration and Optimization

For simplicity and notational conveniency, the presentation given herein assumes integration over the total range of the Euclidean space while the general theory deals with more abstract spaces in a measure theoretic fashion, see [4] for such a presentation. One example where integration is performed over non-Euclidean space is in Bayesian model selection problems where parameters in the space $\cup_k \mathbb{R}^k$ are considered.

6.1.1 Integration

Numerical integration deals with the problem of numerically evaluating general integrals,

$$I = \int_{\mathbb{R}^n} g(x)\, dx. \tag{6.1}$$

Monte Carlo methods for numerical integration regard problems on the form

$$I = \int_{\mathbb{R}^n} f(x)\pi(x)\, dx, \tag{6.2}$$

where $\pi(x)$ is a positive function that integrates to unity,

$$\pi(x) \geq 0, \qquad \int_{\mathbb{R}^n} \pi(x)\, dx = 1.$$

Most problems on the form (6.1) can be transformed into an integral for Monte Carlo evaluation through a suitable factorization of the integrand $g(x) = f(x)\pi(x)$. The assumptions on the factor $\pi(x)$ impose a natural interpretation of $\pi(x)$ as a probability density function. In a Bayesian context, the density of interest is the posterior density of the parameters given the observed data, i.e., $\pi(x) = p(x \,|\, y)$, and considering, e.g., the mean square estimate, we identify $f(x) = x$, and $I = \hat{x}_{\mathrm{MS}}$, in (6.2).

The Monte Carlo methods rely on the assumption that it is possible to draw $N \gg 1$ samples $\{x_i\}_{i=1}^N$ distributed according the probability density $\pi(x)$. Algorithms that achieve this for general classes of distributions $\pi(x)$ are presented in Section 6.2 and Section 6.3. The Monte Carlo estimate of the integral (6.2) is formed by taking the average over the set of samples

$$f_N = \frac{1}{N} \sum_{i=1}^{N} f(x_i) \tag{6.3}$$

where N is assumed to be large. If the samples in the set $\{x_i\}_{i=1}^N$ are independent, f_N will be an unbiased estimate and will almost surely converge to I,

$$\Pr\left(\lim_{N\to\infty} f_N = I \right) = 1 \tag{6.4}$$

by the strong law of large numbers. Moreover, if the variance of $f(x)$,

$$\sigma^2 = \int_{\mathbb{R}^n} (f(x) - I)^2 \pi(x)\, dx = \int_{\mathbb{R}^n} f^2(x)\pi(x)\, dx - I^2 \tag{6.5}$$

is finite, the central limit theorem yields convergence in distribution of the error

$$\lim_{N\to\infty} \sqrt{N}(f_N - I) \sim \mathcal{N}(0, \sigma^2). \tag{6.6}$$

Note that the notation $\mathcal{N}(\mu, P)$ is used for the Gaussian distribution, while $\mathrm{N}(x; \mu, P)$ is the Gaussian density, regarded as a function of the indeterminate x. Even in cases when the samples in the set $\{x_i\}_{i=1}^N$ are dependent it is possible to obtain a law of large numbers and a central limit theorem under weak assumptions. Details are given in [148], but also further discussed in Section 6.3.

The convergence results (6.4) and (6.6) are asymptotic. This means that as $N \to \infty$ we know that the error of the approximation will tend to zero. With support from this asymptotic result we usually assume that a large but finite N will lead to a small error. In practical applications the number of samples might have to be very large for a given error bound. The Monte Carlo methods may therefore be regarded as brute force algorithms of the type presented in Chapter 5. There are, however, two main advantages of Monte Carlo integration compared to straightforward numerical integration. The methods in Chapter 5 generally suffer from intractable demands for computational resources and implementational complexity when applied in high dimensional spaces. The expression (6.6), on the other hand, yields that the error $\varepsilon = f_N - I$ of the Monte Carlo estimate is of the order

$$\varepsilon = O(N^{-1/2}), \tag{6.7}$$

independently of the state dimension, n. Moreover, while the numerical integration methods require the user to define a grid over the integration area that naturally is dependent of the integrand, the estimate (6.3) is obtained using the same technique for any function $f(x)$. One should note though, that even if the error will tend to zero asymptotically in N at a rate that is independent of the state dimension, the constant factor hidden behind the expression (6.7) usually will depend on the state dimension. A given bound on the error ε will often demand more samples in high dimensional problems than in low dimensional ones.

The origin of the positive effects of Monte Carlo integration can be found in the fact that for these methods, the set of samples $\{x_i\}_{i=1}^N$ is automatically chosen in the parts of the state space that are important for the integration result. Since the samples are chosen according to the density $\pi(x)$, which is a factor of the integrand, one can conclude that the effectiveness of the method depend on how the factorization $g(x) = f(x)\pi(x)$ is performed. The more informative, or varying, $\pi(x)$ is compared to $f(x)$, the better will this automatic choice of sample locations be. This claim is verified by (6.5) since the relative smoothness of $f(x)$ compared to $\pi(x)$ determines the size of the variance σ^2. This variance directly affects the size of the error through (6.6). Assuming a positive integrand, the ultimate choice naturally consists of having $\pi(x)$ as the complete integrand, i.e., $f(x) = \alpha$. This would yield $\sigma^2 = 0$, but the estimated integral is then of course known analytically. Usually, however, there is no choice of the factorization of the integrand. In the case of Bayesian estimation, it is natural to consider integration with respect to the posterior density, i.e., $\pi(x) = p(x \,|\, y)$ and no other option is in general available. However, reduction of the variance (6.5), and thereby reduction of the estimation error, can be achieved in several ways, see [45, 46]. The techniques for *variance reduction* usually rely on approximation of the integrand by functions that may

be handled analytically. Therefore, these methods may not be applicable when the dimension is high or the integrand has no closed form analytical expression, see [45, 46].

The numerical integration methods generally approximate the integral by a summation over a grid of regular discretization manually chosen at the support set of the integrand. The Monte Carlo methods utilize the fact that, under the assumption that it is possible to generate N samples from a density given as a factor of the integrand, an adaptive grid well suited for integral approximation is, more or less, automatically obtained. This, in a sense, is the way these methods beat the curse of dimensionality and is the core difference between straightforward numerical integration and Monte Carlo integration methods.

Actually, the efficiency of Monte Carlo integration can be raised beyond (6.7), and an error which asymptotically is inversely proportional to N achieved. This is obtained by using, not independent random variables in the set $\{x_i\}_{i=1}^N$, but choosing each new sample x_i sufficiently far away from the former ones, so that any potential clustering of the samples in the sum (6.3) is alleviated. The means for obtaining such pseudo random variables rely on the theory of equidistant sequences [45, 46], and these methods are sometimes labeled quasi Monte Carlo methods [118]. However, generating equidistant sequences with distribution according to an arbitrary density $\pi(x)$ is in general rather difficult. Moreover, the quasi Monte Carlo algorithms often show an initial error of the order given by (6.7) and only for very large N an error of $O(N^{-1})$ is obtained, see Fearnhead [66].

6.1.2 Optimization

Considering Bayesian MAP estimators, the sought estimate is the location of the maximum peak of the posterior, i.e., the mode of the density. Occasionally, it is only some elements of the posterior parameter vector that are interesting. Then, the location of the maximum peak of one of the marginals of the density is sought. The framework of Monte Carlo integration covers this kind of optimization as well. The MAP estimate takes the form

$$\arg\max_x q(x) \tag{6.8}$$

where we require $q(x) \propto p(x \,|\, y)$. In a Monte Carlo framework, samples $\{x_i\}_{i=1}^N$ distributed according to the, possibly unnormalized, density $q(x)$ may be generated. With high probability, these samples will naturally be located in the areas of the state space where $q(x)$ is large. Hence, a straightforward maximization can be performed with respect to the sample set,

$$\arg\max_{\{x_i\}_{i=1}^N} q(x_i).$$

As N increases it will be more and more probable to find the mode in the set $\{x_i\}_{i=1}^N$. However, this optimization method requires that the function to maximize can be evaluated, at least up to a normalizing factor. When this is intractable, one

has to resort to maximization of the Monte Carlo estimate of the density, i.e., the histogram of the Monte Carlo samples. This will yield a discretization of the state space and thus require even higher values of N for reliable results.

A more appropriate way to solve (6.8) would be to sample from a distribution with support at the global maxima of $q(x)$. One way to achieve this approximately is by *simulated annealing* techniques [150]. After each new generation of a sample point x_i, the distribution generating the samples is altered so that it concentrates successively more and more on the set of global maxima of the distribution $q(x)$. For Bayesian MAP estimation the density that generates sample number i is then chosen according to a distribution $q_i(x)$ satisfying

$$q_i(x) \propto p(x \,|\, y)^{\frac{1}{T_i}}$$

where T_i is a cooling sequence, satisfying $T_{i+1} \leq T_i$ and $\lim_{i\to\infty} T_i = 0$. Under weak regularity assumptions on $p(x \,|\, y)$, the limiting density $q_\infty(x)$ will be a probability measure concentrated on the set of global maxima of $q(x)$, see [81, 92].

6.2 Classical Methods of Sampling

The Monte Carlo framework for numerical integration and optimization rests on the assumption that $N \gg 1$ samples from a generic density $\pi(x)$, are easily obtained in practice. For standard distributions such as uniform, Gaussian, Gamma, Student etc. several perfect random sampling algorithms exist. Uniform i.i.d. random variables can be generated by some pseudo random sequence with very long repetition time, see [126]. Other standard distributions are generally obtained by feeding a, possibly approximative but often exact, inverse of the cumulative distribution function with a pseudo random sequence. Higher dimensional random variables and more general distributions can be generated by combinations and mixtures of basic distributions, see Ripley [126] for a thorough treatment of random number generation. A short summary of classical random number generators is also given in Robert [127, Appendix B].

Usually, the density under consideration is not a familiar combination or mixture of the basic distributions, and it is not possible to directly generate samples from $\pi(x)$. When there is a known upper bound on the density function values, and it is possible to evaluate the density pointwise, it is still possible to generate samples from $\pi(x)$. The rejection sampling procedure, presented in the next subsection, attains this although perhaps rather inefficiently. When it is possible to generate samples from a density similar to the desired one, a correct weighting of the sample set makes Monte Carlo estimation possible. The importance sampling methods of Section 6.2.2 assumes that it is possible to evaluate $\pi(x)$ up to a normalization constant, and that the *proposal distribution* covers the support of $\pi(x)$.

6.2.1 Rejection Sampling

When an upper bound on the range of the generic density function $\pi(x)$ is known and it is possible to evaluate $\pi(x)$ everywhere up to a normalizing constant, a simple rejection procedure can be applied. Let $q(x)$ be a *proposal distribution* from which samples are easily generated, and assume that there exists a known constant $M < \infty$ such that $\pi(x) \leq Mq(x)$ for every $x \in \mathbb{R}^n$. The procedure is to draw a candidate sample x' from $q(x)$ and accept it with probability $\frac{\pi(x')}{Mq(x')}$. If x' is rejected, the procedure continues to draw samples from $q(x)$ until an accepted sample is obtained. The finally accepted candidate will be an exact draw from $\pi(x)$.

Algorithm 6.1 (Rejection Sampling)

1. Sample $x' \sim q(x)$ and $u \sim \mathcal{U}(0,1)$.
2. If $u < \frac{\pi(x')}{Mq(x)'}$ return x, otherwise goto step 1.

To see that this algorithm actually generates samples from $\pi(x)$ consider the scalar case, and study a generic candidate draw x'

$$\Pr(x' \leq t, \text{and } x' \text{ is accepted}) = \int_{-\infty}^{t} \frac{\pi(x)}{Mq(x)} q(x)\, dx = \frac{1}{M} \int_{-\infty}^{t} \pi(x)\, dx.$$

The acceptance probability of a generic x' is found by setting $t = \infty$,

$$\Pr(x' \text{ is accepted}) = \int_{\mathbb{R}} \frac{\pi(x)}{Mq(x)} q(x)\, dx = \frac{1}{M}. \tag{6.9}$$

Hence,

$$\Pr(x' \leq t \,|\, x' \text{ is accepted}) = \int_{-\infty}^{t} \pi(x)\, dx$$

which shows that the accepted x' is exactly a draw from $\pi(x)$. However, to bound $\pi(x)$ over the complete state space is in many cases impossible. If a bounding function $q(x)$ can be found, the constant M may need to be very large and thus will lead to an inefficient algorithm since the probability of accepting a candidate draw (6.9) will be low.

In a Bayesian framework, the normalizing constant of the desired posterior density is generally unknown,

$$\pi(x) = p(x \,|\, y) \propto p(y \,|\, x)p(x)$$

and difficulties to evaluate the acceptance rate of the candidate draw in Algorithm 6.1 may arise. Fortunately, the rejection procedure can be applied by incorporating the unknown constant into the average acceptance probability (6.9).

Assume that global bound on the likelihood $p(y \mid x) \leq C$ for all $x \in \mathbb{R}^n$ is known, and use, e.g., the prior as proposal distribution $q(x) = p(x)$. Since the bound on the posterior density is

$$\pi(x) = \frac{p(y \mid x)p(x)}{p(y)} \leq \frac{Cp(x)}{p(y)} = Mp(x),$$

the acceptance probability for a specific candidate draw x becomes

$$\frac{\pi(x)}{Mq(x)} = \frac{p(x \mid y)p(y)}{Cp(x)} = \frac{p(y \mid x)}{C}.$$

This probability can be computed even though the normalization constant of the posterior density is unknown. The average acceptance probability $\frac{p(y)}{C}$ is, however, not known and can be arbitrarily low depending on the observation.

6.2.2 Importance Sampling

Importance sampling also deals with a proposal distribution $q(x)$ which is easy to generate samples from. However, the only general assumption on the *importance function* $q(x)$ is that its support set covers the support of $\pi(x)$, i.e., that $\pi(x) > 0 \Rightarrow q(x) > 0$ for all $x \in \mathbb{R}^n$. Under this assumption, any integral on the form (6.2) can be rewritten

$$I = \int_{\mathbb{R}^n} f(x)\pi(x)\,dx = \int_{\mathbb{R}^n} f(x)\frac{\pi(x)}{q(x)}q(x)\,dx. \tag{6.10}$$

A Monte Carlo estimate is computed by generating $N \gg 1$ independent samples from $q(x)$, and forming the weighted sum

$$f_N = \frac{1}{N}\sum_{i=1}^{N} f(x_i)w(x_i), \qquad \text{where} \qquad w(x_i) = \frac{\pi(x_i)}{q(x_i)} \tag{6.11}$$

are the *importance weights*. It is straightforward to verify that (6.4) and (6.6) are satisfied for the importance sampling Monte Carlo estimate (6.11).

If the normalizing factor of the target density $\pi(x)$ is unknown, the importance weights in (6.11) can only be evaluated up to a normalizing factor. Then, the weights can be formed using a function proportional to the target density and then normalized afterwards, forming the estimate

$$f_N = \frac{\sum_{i=1}^{N} f(x_i)w(x_i)}{\sum_{j=1}^{N} w(x_j)}, \qquad \text{where} \qquad w(x_i) \propto \frac{\pi(x_i)}{q(x_i)}. \tag{6.12}$$

This technique is practically applied in the Bayesian framework and therefore often referred to as *Bayesian importance sampling*. The estimate (6.12) is biased for finite N, but asymptotically both a law of large numbers and a central limit theorem hold.

Theorem 6.1 (Geweke [72])
When (6.10) *exists and is finite, the estimate* (6.12) *converges almost everywhere*

$$\Pr\Big(\lim_{N\to\infty} f_N = I\Big) = 1.$$

Additionally, if $\mathsf{E}(w(x)) < \infty$ *and* $\mathsf{E}\big(f^2(x)w(x)\big) < \infty$,

$$\lim_{N\to\infty} \sqrt{N}(f_N - I) \sim \mathcal{N}(0, \sigma^2) \tag{6.13}$$

where $\sigma^2 = \mathsf{E}\big((f(x) - I)^2 w(x)\big)$. *All expectations above are performed w.r.t. the density* $\pi(x)$.

Geweke [72] also gives a sufficient condition for satisfying the second assumption in the theorem above. Geweke states that (6.13) holds if $w(x) < C < \infty$ for any $x \in \mathbb{R}^n$ and the variance of $f(x)$, w.r.t. $\pi(x)$, is finite. In the Bayesian framework,

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

The prior $p(x)$ will suffice as a choice for importance distribution yielding that the unnormalized weights are given by $w(x) = p(y \,|\, x)$. With a bounded likelihood and some weak regularity conditions on $\mathsf{E}_\pi\, f(x)$ and $\mathsf{E}_\pi\, f^2(x)$, the Bayesian importance sampling procedure will thus yield asymptotically consistent estimates of (6.10). A more cleverly chosen importance function would depend on the observation.

The importance sampling procedure yields a Monte Carlo estimate of the integral (6.10). The Sampling Importance Resampling (SIR) algorithm of Rubin [129] is a procedure for generating an approximately independent draw from $\pi(x)$ using its weighted approximation. The independent draw from $\pi(x)$ is obtained by inserting a resampling step in the spirit of Effron's Bootstrap [121] after the weight calculations.

Algorithm 6.2 (Sampling Importance Resampling, SIR)

1. Generate M independent samples $\{x_i\}_{i=1}^M$ with common distribution $q(x)$.
2. Compute the weight $w_i \propto \pi(x_i)/q(x_i)$ for each x_i.
3. Normalize the weights $w_i := \gamma^{-1} w_i$, where $\gamma = \sum_{j=1}^M w_j$.
4. Resample with replacement N times from the discrete set $\{x_i\}_{i=1}^M$ where $\Pr(\text{resampling } x_i) = w_i$.

For reliable results, M should be chosen greater than N. The guideline of Rubin [129] is to choose at least $M = 10N$. With the SIR procedure, an independent draw from an approximation of $\pi(x)$ is produced. Only asymptotically, as $M \to \infty$, will a draw from $\pi(x)$ be obtained. The rejection sampling procedure, on the other hand, yields exact draws from $\pi(x)$, without approximations.

The choice of importance function is crucial for successful application of the technique presented in this subsection. Apart from having a support that covers $\pi(x)$, the general suggestion found in most works applying importance sampling is to choose an importance function that mimics the density $\pi(x)$ but has thicker tails. A justification of this choice is that with $\pi(x)$ as importance function one may need a very large N to get any support in its tails. It is thus better to pull samples more frequently from the tails and downweight them. By tailoring the importance function $q(x)$ to the integrand $f(x)$, the variance in (6.6) can actually be reduced below the one obtained if perfect Monte Carlo simulation was possible, i.e., if $q(x) = \pi(x)$. Thus, importance sampling can also be regarded as a variance reduction scheme for general Monte Carlo integration, as done by Davis and Rabinowitz [45, 46]. However, we will usually be more interested in the weighted approximation of the density itself, and not in a specific integral with respect to the density. These variance reduction techniques are therefore of less interest to us.

6.2.3 Summary

The success of applying either of the rejection sampling or importance sampling methods relies on determining good proposal distributions and importance functions, respectively. A badly chosen proposal distribution yields a low acceptance rate in the rejection sampling algorithm. Likewise, choosing the wrong importance function yields a large variance of the importance weights with only some samples contributing to the sum (6.11), and thus a slow convergence of the estimate. These problems frequently arise in high dimensional problems when the approximate shape of the posterior is unknown. The general guideline is to use these methods when the dimension of the state space is less than 10 [4, 46]. The Markov chain Monte Carlo (MCMC) methods of the following section are a suite of simulation based algorithms that are applicable to much more general and complex models and in particular to higher dimensional problems.

6.3 Markov Chain Monte Carlo

The MCMC algorithms are iterative procedures that output a sequence of random samples by simulating a Markov chain tailored to have a limit distribution given by the density $\pi(x)$. By discarding an initial *burn in* phase of the Markov chain, ergodic averages of the chain realization can be used to estimate integrals with respect to $\pi(x)$.

There are several references that treat the topic of Markov chain Monte Carlo estimation with greater depth and in more detail than the presentation given here. A detailed, yet compact, review over MCMC methods with signal processing applications is provided by Andrieu et al. [4]. A comprehensive set of separate articles covering a wide variety of both theoretical and practical guidelines towards applying MCMC methods is found in the volume edited by Gilks et al. [73]. The survey article of Tierney [148] presents comprehensibly the theoretical foundations of Markov

chain Monte Carlo methods. The presentation that follows below only considers the case with Markov chains defined over Euclidean space $\mathbb{R}^n$. Moreover, we will assume all random variables $\boldsymbol{x}$ to have well defined probability density functions $p(x) < \infty$ for all $x \in \mathbb{R}^n$. A more thorough presentation in a measure-theoretic framework is given in either of [4, 148, 149].

6.3.1 Theory

A Markov chain is a sequence of random variables $\{x_t\}_{t \geq 0}$ such that

$$\Pr(x_t \in A \,|\, x_0, \ldots, x_{t-1}) = \Pr(x_t \in A \,|\, x_{t-1})$$

for all $A \subset \mathbb{R}^n$. The *transition kernel* of the Markov chain is the conditional density function

$$K(x_{t-1}, x_t) \triangleq p(x_t \,|\, x_{t-1}). \tag{6.14}$$

A time-homogenous Markov chain is one where the transition kernel is explicitly independent of the time index t. In the case of Markov chains over discrete state spaces, the transition kernel (6.14) is a discrete transition probability matrix. The p-step transition kernel is given by

$$K_p(x_{t-p}, x_t) = p(x_t \,|\, x_{t-p}).$$

The initial distribution of the Markov chain is $p(x_0)$ and may, in the general case, be a Dirac delta measure indicating that the initial state of the Markov chain is deterministic.

The idea behind Markov chain Monte Carlo methods is to construct a transition kernel such that the limiting, or stationary, distribution of the output from the Markov chain is some desired probability density function $\pi(x)$. In order to fulfill this requirement, a condition of invariance must hold between the transition kernel $K(x_{t-1}, x_t)$ of the Markov chain and $\pi(x)$.

Definition 6.1
The probability density function $\pi(x)$ is said to be invariant *(or* stationary*) with respect to the transition kernel K if*

$$\pi(x) = \int_{\mathbb{R}^n} K(x_{t-1}, x)\pi(x_{t-1})\, dx_{t-1}$$

for all $x \in \mathbb{R}^n$.

The density $\pi(x)$ being invariant with respect to the Markov chain implies that if $x_t \sim \pi(\cdot)$ for some t, the output of the chain will remain marginally distributed according to $\pi(\cdot)$ for all future time instants. A sufficient condition to ensure π-invariance is to assure that the Markov chain is π-reversible.

Definition 6.2
A transition kernel K is π-reversible if, it satisfies

$$K(x,y)\pi(x) = K(y,x)\pi(y). \tag{6.15}$$

Condition (6.15) is called the *detailed balance condition*, weaker conditions for reversibility can be defined, e.g., in [148]. Generally, the reversibility condition should say that the probability of the Markov chain moving from a region A to a region B is equal to the probability of moving from B to A. This should hold whenever the state is in the stationary regime, i.e., under the assumption that it is distributed according to $\pi(x)$ before the move takes place.

Most MCMC algorithms are π-reversible by construction, and therefore $\pi(x)$ is an invariant distribution of the Markov chain. Hence, the reversibility condition does often not need to be verified in practical application of MCMC algorithms. Suppose that we know how to construct a Markov chain with transition kernel K having $\pi(x)$ as its invariant density. In order to apply this Markov chain to Monte Carlo integration we need to assure that the sample path average of the output from this chain

$$f_N = \frac{1}{N}\sum_{i=1}^{N} f(x_i), \tag{6.16}$$

possibly discarding a burn-in phase, converges to the expectation $\mathsf{E}(f(x))$ when it exists. Since the initial distribution $p(x_0) \neq \pi(x_0)$ in general, convergence should be attained for any suitable initial distribution $p(x_0)$. A minimal requirement for this is that all interesting parts of the state space can be reached by the Markov chain.

Definition 6.3
A Markov chain is φ-irreducible if for any $A \subset \mathbb{R}^n$

$$\int_A \varphi(y)\,dy > 0 \quad \Rightarrow \quad \exists p \in \mathbb{N} \text{ such that } \int_A K_p(x,y)\,dy > 0$$

for any $x \in \mathbb{R}^n$.

Irreducibility defines the regions of the state space which the chain can move around in, but never leave. A sufficient condition for a kernel K to be φ-irreducible is that for some $p \geq 1$ the kernel K_p can be factorized by $\varphi(y)$, i.e., that there exists a positive function $f(x,y) > 0$ such that $K_p(x,y) = f(x,y)\varphi(y)$, see Tierney [149]. If a chain is irreducible with respect to some density φ and has invariant density π, then the chain is π-irreducible according to Tierney [149]. This leads to the existence of a strong law of large numbers for Markov chain Monte Carlo methods.

Theorem 6.2 (Theorem 4.3 of Tierney [149, p. 65])
Let $\{x_t\}_{t=0}^{N}$ be a π-irreducible Markov chain with transition kernel K having invariant density π. Let $f : \mathbb{R}^n \to \mathbb{R}$ be such that $\mathsf{E}_\pi(f(x)) < \infty$. Then

$$\Pr\left(\lim_{N\to\infty} f_N = \mathsf{E}_\pi(f(x))\right) = 1$$

where f_N is the Monte Carlo estimate (6.16). *The claim holds for chains initialized at π-almost all x_0.*

As long as the chain is initialized in the support set of $\pi(x)$, the theorem says that the average proportion of time spent in any region $A \subset \mathbb{R}^n$ will converge to $\pi(A)$. Additional assumptions on the Markov chain can be used to alleviate the restriction of the initial states, see [148, 149].

Stronger results on the limiting distribution of the chain can be obtained by restricting the chain from exploring the state space in a periodic manner. The chain is then said to be *aperiodic*, and Theorem 4.4 of Tierney [149, p. 65] gives that the transition kernel $K_N(x_0, \cdot)$ tends to $\pi(\cdot)$ in total variation norm as N tends to infinity, for all x_0 in the support set of $\pi(x)$. Practically, the main interest lies in the convergence of the sample path average (6.16) rather than assuring the limiting distribution of the chain, and therefore the demand for an aperiodic kernel can often be relieved. Still, many MCMC algorithms are aperiodic by construction. This includes the large class of Metropolis–Hastings algorithms presented in Section 6.3.2.

The simulated output from the Markov chain is a statistically dependent set of samples where the marginal distribution of the output approaches $\pi(x)$ under the invariance, irreducibility and aperiodicity conditions. Contrary to the central limit theorem result (6.6) for the case of estimates (6.16) based on independent sets of samples, the MCMC convergence results cited above do not give any information about the rate of convergence towards the invariant distribution. In order to establish such bounds, the issue of ergodicity of the Markov chain must be addressed. A central limit theorem can be established if the chain is uniformly ergodic and the variance of $f(x)$ is finite. Then, the dimension independent error rate (6.7) holds for the sample path averages of MCMC algorithms, see [148, 149].

To summarize this section, a Markov chain having $\pi(x)$ as a stationary density being aperiodic and irreducible will asymptotically generate an output with distribution $\pi(x)$. The invariance with respect to $\pi(x)$ is usually assured by the MCMC procedure from the construction of the transition kernel, while irreducibility and aperiodicity need to be investigated for each application. In summary, the irreducibility condition assures that all areas in the support set of $\pi(x)$ may be reached by the chain, and the aperiodicity condition that the kernel does not exhibit any periodic behavior that would induce undesirable dependencies in the chain output.

6.3.2 Algorithms

Most algorithms for Markov chain Monte Carlo estimation are based on the algorithm of Hastings [85], which is a generalization of the algorithm of Metropolis et al. [116]. The Metropolis–Hastings algorithm resembles the previously described algorithms of this chapter in that a proposal distribution is used to generate the samples. However, the output of the algorithm is a Markov chain so the proposal density may depend on the current state of the chain. Let x denote the current

state of the chain, during each iteration of the Metropolis–Hastings algorithm a candidate sample z is drawn from the proposal $q(z \mid x)$ and accepted with a probability given by

$$\alpha(x, z) = \min\left(1, \frac{\pi(z)q(x \mid z)}{\pi(x)q(z \mid x)}\right). \tag{6.17}$$

If the candidate is accepted the chain moves to the new position, while a rejection of the candidate leaves the chain at the current position in the state space. An interpretation of (6.17) is that all candidates that yield an increase of $\pi(\cdot)$, and is not too unlikely to return from, are accepted.

Algorithm 6.3 (Metropolis–Hastings)

1. Initialize by setting $t = 0$ and choosing x_0 randomly or deterministically.
2. Sample $z \sim q(z \mid x_t)$.
3. Sample $u \sim \mathcal{U}(0, 1)$.
4. Compute the acceptance probability $\alpha(x_t, z)$ in (6.17).
5. If $u \leq \alpha(x_t, z)$ accept the move and set $x_{t+1} = z$. Otherwise set $x_{t+1} = x_t$.
6. Increase t and return to item 2.

One very important feature of the Metropolis–Hastings algorithm is that the distributions $\pi(x)$ only need to be known up to a normalizing constant. The normalizing factor of $\pi(x)$ cancels in the expression for the acceptance probability (6.17), which thus may be evaluated even if $\int \pi(x)\, dx$ is unknown.

Example 6.1
Consider the general case of Bayesian estimation where

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

After generating a candidate sample from a suitable $q(z \mid x)$, this sample is accepted with probability

$$\alpha(x, z) = \min\left(1, \frac{p(y \mid z)p(z)q(x \mid z)}{p(y \mid x)p(x)q(z \mid x)}\right).$$

Using the prior as proposal distribution $q(z \mid x) = p(z)$, the acceptance probability simplifies to

$$\alpha(x, z) = \min\left(1, \frac{p(y \mid z)}{p(y \mid x)}\right).$$

And thus the candidate is always accepted if it increases the likelihood.

By construction, an MCMC algorithm should ensure the desired density $\pi(x)$ to be invariant with respect to the kernel of the chain. To verify that the Metropolis–Hastings algorithm satisfies this assumption, study the transition kernel

$$p(x_{t+1} \,|\, x_t) = q(x_{t+1} \,|\, x_t) \Pr(\text{accepting } x_{t+1} \,|\, x_t) + \delta(x_{t+1} - x_t) \Pr(\text{rejecting } x_{t+1} \,|\, x_t),$$

which straightforwardly can be written

$$p(x_{t+1} \,|\, x_t) = q(x_{t+1} \,|\, x_t)\alpha(x_t, x_{t+1}) + \delta(x_{t+1} - x_t) \left(1 - \int q(z \,|\, x_t)\alpha(x_t, z)\, dz\right). \quad (6.18)$$

The acceptance probability (6.17) yields the detailed balance relation

$$\pi(x_t)q(x_{t+1} \,|\, x_t)\alpha(x_t, x_{t+1}) = \pi(x_{t+1})q(x_t \,|\, x_{t+1})\alpha(x_{t+1}, x_t).$$

Inserting this into (6.18) we obtain

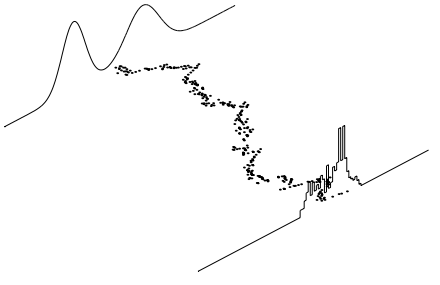
$$\pi(x_t)p(x_{t+1} \,|\, x_t) = \pi(x_{t+1})p(x_t \,|\, x_{t+1})$$

which yields that the kernel of the Metropolis–Hastings algorithm is π-reversible for almost any choice of $q(z \,|\, x)$. Note that this only proves that $\pi(x)$ is a stationary distribution of the Metropolis–Hastings algorithm. For the sample path average to actually converge, and for the samples from the chain to converge in distribution, additional requirements of irreducibility and aperiodicity must be justified. A simple condition that ensures these properties is that the proposal $q(\cdot \,|\, x)$ is continuous and strictly positive on the support of $\pi(\cdot)$ for any x, see [4]. Details regarding Metropolis–Hastings convergence can be found in Roberts [128].

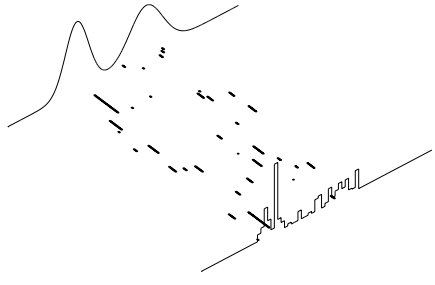
Even though the Metropolis–Hastings algorithm will be π-invariant for many choices of $q(z \,|\, x)$, the choice of proposal distribution will naturally affect the convergence of the chain.

Example 6.2

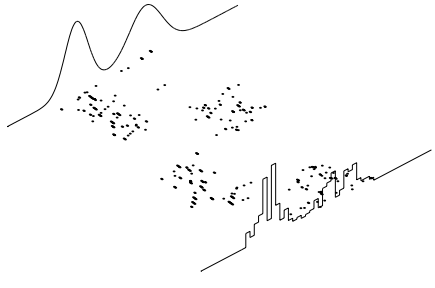
Figure 6.1 illustrates the effect of choosing different proposal distributions to sample from a Gaussian mixture using the Metropolis–Hastings algorithm. The chain is deterministically initialized between the modes of $\pi(x)$, and the proposal $q(z \,|\, x)$ yields a random walk, i.e., the proposal point is chosen as an independent zero mean addition to the current state of the chain. Different behavior is obtained depending on the average size of the steps proposed by $q(z \,|\, x)$. With too small steps the chain gets stuck around a local mode of $\pi(x)$ and with too large steps the proposal will often end up in the tails of $\pi(x)$ and thus frequently be rejected by the Metropolis–Hastings algorithm. The cases in Figure 6.1(a) and 6.1(b) are often referred to as slowly mixing *chains, while Figure 6.1(c) shows a choice of proposal yielding a good mixing of the chain.*



(a) $\sigma = 0.1$, only one mode of $\pi(x)$ is explored.



(b) $\sigma = 20$, many candidates are rejected.



(c) $\sigma = 2$, both modes are visited while the acceptance probability still is high.

Figure 6.1: Sampling from a Gaussian mixture using Metropolis–Hastings algorithm with Gaussian random walk proposal $q(z \mid x_t) = \mathrm{N}(z; x_t, \sigma^2)$. The plots show $\pi(x)$, 300 samples generated by the algorithm and a histogram over the chain output for different choices of σ.

It is obvious from this example that the choice of proposal distribution determines the efficiency of the algorithm. Apart from choosing different proposal densities, several other approaches to updating the chain have been presented in the literature.

Common proposal choices

A simplistic way to choose the proposal is to have it fixed, and independent of the current state of the chain. The *independence sampler* [148] with a proposal distribution $q(z \mid x) = q(z)$ yields an acceptance probability (6.17) of

$$\alpha(x, z) = \min\left(1, \frac{w(z)}{w(x)}\right) \qquad \text{where} \qquad w(x) = \frac{\pi(x)}{q(x)},$$

which is the type of proposal that was used in Example 6.1.

In the original algorithm of Metropolis et al. [116], symmetric proposals are considered, i.e., proposal distributions such that $q(x \mid z) = q(z \mid x)$. The acceptance probability then simplifies to

$$\alpha(x, z) = \min\left(1, \frac{\pi(z)}{\pi(x)}\right).$$

Example 6.2 shows an example of such a proposal choice.

Blocking

An alternative way to propose a new candidate vector z is to update scalar or low dimensional subcomponents of x, in a *blocking* scheme. This was actually the framework proposed for MCMC methods by Metropolis et al. [116], and is often referred to as *single-component*, or *one-at-a-time* Metropolis–Hastings. This can be an particularly efficient approach in high dimensional problems where it often is hard to choose good proposal distributions.

In single-component Metropolis–Hastings each component of x is updated according to a Metropolis–Hastings step where the invariant distribution is the *full conditional* distribution of that component. The full conditional distribution for entry number i is

$$\pi(x_i \mid x_{\neg i}) = \frac{\pi(x)}{\int_{\mathbb{R}} \pi(x)\, dx_i}$$

where $x_{\neg i}$ is the vector consisting of all entries of x except for entry number i,

$$x_{\neg i} = [x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n]^T.$$

A unique proposal $q_i(z_i \mid x_i, x_{\neg i})$ can be used for each entry i. The algorithm cycles through the entries of x sampling from each proposal and accepts the candidate entry z_i with probability

$$\alpha(x_{\neg i}, x_i, z_i) = \min\left(1, \frac{\pi(z_i \mid x_{\neg i}) q_i(x_i \mid z_i, x_{\neg i})}{\pi(x_i \mid x_{\neg i}) q_i(z_i \mid x_i, x_{\neg i})}\right). \tag{6.19}$$

The newly accepted or rejected entry is then inserted into x and the next candidate component is sampled from the proposal distribution of that entry.

Gibbs sampling

The Gibbs sampling algorithm by Geman and Geman [71] is the most commonly applied MCMC algorithm. The Gibbs sampling algorithm can be seen as a blocking Metropolis–Hastings procedure where proposal samples are drawn directly from the full conditional distributions. Inserting

$$q_i(z_i \mid x_{\neg i}) = \pi(z_i \mid x_{\neg i})$$

into (6.19) yields an acceptance probability of one. Hence, all candidates are accepted and no acceptance probability has to be evaluated.

Algorithm 6.4 (Gibbs sampling)

1. Initialize by setting $t = 0$ and choose $x^{(0)}$ randomly or deterministically.
2. Cycle through the entries of x and sample from the full conditionals,
   - $x_1^{(t)} \sim \pi(x_1 \mid x_{\neg 1}^{(t)})$
   - $x_2^{(t)} \sim \pi(x_2 \mid x_{\neg 2}^{(t)})$
   - $\vdots$
   - $x_n^{(t)} \sim \pi(x_n \mid x_{\neg n}^{(t)})$
3. Output $x^{(t)}$, increase t and return to item 2.

Above, $x_{\neg i}^{(t)} \triangleq [x_1^{(t)}, \ldots, x_{i-1}^{(t)}, x_{i+1}^{(t-1)}, \ldots, x_n^{(t-1)}]^T$.

The algorithm presented here is the deterministic version of the Gibbs sampler. Alternatively, one can cycle through the entries of x in a random fashion. Moreover, other partitions of x can be used, e.g., one can choose to sample highly correlated entries as one block. The Gibbs sampler is applied to a measurement association problem in Chapter 8.

6.4 Recursive Monte Carlo Methods

Recursive, or sequential, Monte Carlo methods is the unifying name for those algorithms that deal with the recursive estimation problem using a Monte Carlo integration approach. The main difference between the Monte Carlo algorithms described previously in this chapter and the algorithms for recursive estimation is that the target density for the recursive case is time dependent, and that in general no explicit expression for the density exists. Most algorithms cited above for generating i.i.d. samples from a generic distribution $\pi(x)$ relied on being able

to evaluate $\pi(x)$ exactly for any $x \in \mathbb{R}^n$, at least up to a normalizing factor. This assumption cannot be met for most recursive estimation problems.

In the case of recursive Bayesian estimation the distribution of interest is the posterior filter density, i.e.,

$$\pi_t(x_t) = p(x_t \,|\, \mathbb{Y}_t). \tag{6.20}$$

For each time t, we seek to evaluate integrals with respect to (6.20), e.g., its first and second central moments,

$$\hat{x}_t^{\mathrm{MS}} = \int_{\mathbb{R}^n} x_t \pi_t(x_t)\, dx_t \qquad \text{and} \qquad \int_{\mathbb{R}^n} (x_t - \hat{x}_t^{\mathrm{MS}})(x_t - \hat{x}_t^{\mathrm{MS}})^T \pi_t(x_t)\, dx_t. \tag{6.21}$$

In Theorem 3.3, a conceptual expression for the recursive update of the posterior filter density was given,

$$p(x_t \,|\, \mathbb{Y}_t) \propto p(y_t \,|\, x_t) p(x_t \,|\, \mathbb{Y}_{t-1}) \tag{6.22a}$$

$$p(x_{t+1} \,|\, \mathbb{Y}_t) = \int_{\mathbb{R}^n} p(x_{t+1} \,|\, x_t) p(x_t \,|\, \mathbb{Y}_t)\, dx_t. \tag{6.22b}$$

Above, the integral for computing the normalizing factor of (6.22a) has been hidden behind the proportionality relation $\propto$. It was also noted in Chapter 3 that in general there exists no explicit analytical expression for propagating $p(x_t \,|\, \mathbb{Y}_t)$ through (6.22). Hence, apart from numerically evaluating integrals such as (6.21), the recursive methods must also propagate an approximative description of the posterior density itself, simply because there is no analytical expression for this function. The recursive Monte Carlo methods achieve this by applying Monte Carlo approximation techniques to all the integrals of (6.22) and (6.21). Straightforwardly, performing these approximations yields a recursive propagation of a set of samples approximately drawn from the posterior filter density, and possibly a probability weight assigned to each sample. This cloud, or swarm, of particles adapts and evolves with time and incoming observations y_t, so that the number of particles in each subregion of the state space reflects the probability of finding the true state in that region. Due to this analogy, recursive Monte Carlo methods are sometimes referred to as *particle filters*.

The sequel of this section is devoted to algorithms for recursive Monte Carlo estimation. We present the most commonly applied procedures and give some implementational guidelines for these algorithms. Several surveys over sequential Monte Carlo methods have appeared lately in the literature. The material in this section is mainly based on the compact and comprehensive reviews presented by Doucet [60] and Liu and Chen [110]. Both these references focus on importance sampling methods while the thesis of Fearnhead [66] gives some alternative approaches and more background to the general filtering problem.

6.4.1 Algorithms

In all recursive Monte Carlo filters the posterior density is approximated by a swarm of $N \gg 1$ points $\{x_t^i\}_{i=1}^N$ in the state space $\mathbb{R}^n$. The number of points in

each subregion of $\mathbb{R}^n$ reflects the probability for finding the true state in this region. The algorithms differ in the way this swarm of particles is recursively propagated and affected by the measurements.

The Bayesian Bootstrap Filter

The Bayesian bootstrap filter, or recursive SIR, is an application of the SIR procedure Algorithm 6.2 to the recursive propagation (6.22). The original application of the SIR procedure to recursive estimation by Handschin and Mayne [84], and Handschin [83] focused on computing approximations to the conditional mean and covariance. The Bayesian Bootstrap filter of Gordon et al. [78] applies the same idea but uses the SIR to recursively compute an approximation to the complete posterior distribution. It was the seminal paper of Gordon et al. [78] which, together with the advent of cheap high performing computers, initiated the rapid growth of the research field of Monte Carlo approximation methods for recursive estimation.

The algorithm starts with a prior cloud of particles $\{x_t^i\}_{i=1}^N$ assumed to be an i.i.d. sample from $p(x_t \mid \mathbb{Y}_{t-1})$ such that it can be used to approximate the prior filter density,

$$p(x_t \mid \mathbb{Y}_{t-1}) \approx \frac{1}{N} \sum_{i=1}^{N} \delta(x_t - x_t^i).$$

Inserting this approximation into (6.22a) and applying the SIR procedure, a similar sample from $p(x_t \mid \mathbb{Y}_t)$ is obtained by resampling with replacement as follows. A set of N new samples $\{x_t^{i\star}\}_{i=1}^N$ is produced by resampling with replacement from the set $\{x_t^i\}_{i=1}^N$ where the probability of resampling state x_t^i is proportional to $p(y_t \mid x_t^i)$. After this resampling operation, multiple copies of those samples in $\{x_t^i\}_{i=1}^N$ that correspond to a large likelihood are found in the new set $\{x_t^{i\star}\}_{i=1}^N$, while some of the samples with a comparatively low likelihood are not resampled at all. The resampled set of points is an approximate i.i.d. draw from the filter density $p(x_t \mid \mathbb{Y}_t)$ and hence, applying a Monte Carlo approximation to (6.22b) yields that

$$p(x_{t+1} \mid \mathbb{Y}_t) \approx \frac{1}{N} \sum_{i=1}^{N} p(x_{t+1} \mid x_t^{i\star}). \tag{6.23}$$

A fresh, unweighted, set of N i.i.d. samples from this mixture density is needed in order to close the recursion loop by increasing t and continue updating the particle cloud with the next measurement. The most straightforward and computationally efficient way to produce this set is to generate one sample from each term in the sum (6.23). Hence, a new set $\{x_{t+1}^i\}_{i=1}^N$ is formed by generating N independent draws from $p(x_{t+1} \mid x_t^{i\star})$, one for each i. Through this prediction the resampled copies in $\{x_t^{i\star}\}_{i=1}^N$ will naturally explore the state space independently and assign probability mass to a region in the state space corresponding to the multiply resampled candidate. The prediction step yields an approximately i.i.d. draw from $p(x_{t+1} \mid \mathbb{Y}_t)$ which can be used in the next iteration of the algorithm.

In accordance with the SIR procedure, one can choose to base the resampling on a set of $M = rN$ samples for some $r \in \mathbb{N}$. An algorithmic summary of the recursive SIR procedure follows.

Algorithm 6.5 (Recursive SIR, or Bayesian Bootstrap)

1. Set $t = 0$, and generate M samples $\{x_0^i\}_{i=1}^M$ from $p(x_0)$.
2. Compute the likelihood weights $w_i = p(y_t \,|\, x_t^i)$ for $i = 1, \ldots, M$.
3. Normalize the weights $w_i := \gamma^{-1} w_i$, where $\gamma = \sum_{j=1}^M w_j$.
4. Generate a new set $\{x_t^{i\star}\}_{i=1}^N$ by resampling with replacement N times from the discrete set $\{x_t^j\}_{j=1}^M$ where $\Pr(x_t^{i\star} = x_t^j) = w_j$.
5. Predict each of the resampled states independently r times. Thus, generating the set $\{x_{t+1}^j\}_{j=1}^M$, where

$$x_{t+1}^{(i-1)r+k} \sim p(x_{t+1} \,|\, x_t^{i\star}) \qquad \text{for all } k = 1, \ldots, r \text{ and } i = 1, \ldots, N.$$

6. Increase t and iterate to item 2.

The mean-square error estimate and its estimation error covariance is given by importance weighted Monte Carlo estimates of (6.21). In Algorithm 6.5 the calculations

$$\hat{x}_t^{\mathrm{MS}} = \sum_{i=1}^N w_i x_t^i \qquad \text{and} \qquad P_t = \sum_{i=1}^N w_i (x_t^i - \hat{x}_t^{\mathrm{MS}})(x_t^i - \hat{x}_t^{\mathrm{MS}})^T$$

are inserted between items 3 and 4. In the original presentation, Gordon et al. [78] actually choose to calculate the estimate after the resampling step. If the factor between M and N is large, this will considerably decrease the computational burden.

Figure 6.2 shows an illustration of one iteration of the recursive SIR procedure. With a flat prior at time t, the samples are uniformly distributed over the state space. Evaluating the likelihood on this set and resampling with respect to this weight, the dark circles indicate the size of the likelihood and hence reflects the average number of resampled offsprings. The offsprings are predicted forward in time and it is illustrated how they explore the state space in such a manner that the resulting set of points gives a good approximation to the posterior.

The main advantages of Algorithm 6.5 is its ease of implementation and its applicability to a very large class of recursive estimation problems. The mild assumptions for applying this algorithm are that i.i.d. samples from $p(x_0)$ and from $p(x_{t+1} \,|\, x_t)$ are easy to generate, and that it is possible to evaluate likelihood $p(y_t \,|\, x_t)$ for any y_t and x_t. However, problems do occur, e.g., in fixed parameter estimation when the state transition density $p(x_{t+1} \,|\, x_t)$ is singular. Then the

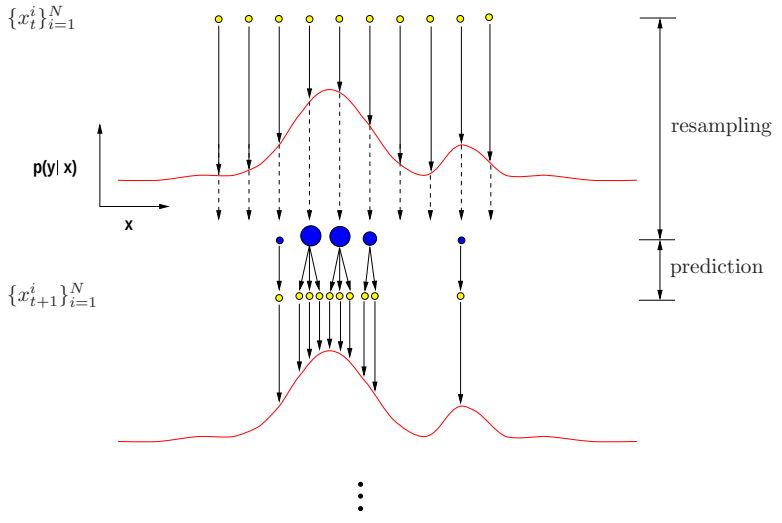


Figure 6.2: Illustration of the recursive SIR procedure. The uniformly distributed prior samples are resampled according the their likelihood value. In the prediction step they diffuse independently and the resulting sample set gives an approximate description of the posterior. Modified from Figure 1 of de Freitas et al. [48], used with permission.

multiple copies after the resampling step will not explore the state space but remain at the same positions. The problem arises from the fact that the discrete set $\{x_t^i\}_{i=1}^M$ is not representable for the prior density, and therefore can be alleviated by increasing M. Alternatively, a kernel density estimation step can be inserted.

The recursive SIR procedure summarized in Algorithm 6.5 is a fundamental version of several similar SIR procedures for recursive estimation. These are mainly enhancements of the original algorithm, developed to overcome some of its weaknesses. When fixed parameters are estimated recursively, the information about them can only increase with time. With particle filters such as the Bayesian bootstrap, this will have the effect that after some iterations only one candidate state vector value will be present in the particle set. Extreme outlier observations can render the same type of behavior. Either the outlier is due to a large measurement error or it is a true outlier with valuable state information. Regardless of the origin of the outlier, the likelihood of the outlying measurement will have its main support on the tail of the prior distribution. This tail is represented by only a few samples of the particle cloud. The resampling step will therefore severely impoverish the particle representation of the posterior. Increasing the number of samples representing the prior by choosing a large value of r in Algorithm 6.5 will alleviate these effects. This idea is sensible even in the non-recursive case and was proposed in the original SIR procedure by Rubin [129]. The technique is some-

times referred to as prior boosting. The article of Gordon et al. [78], where the Bayesian bootstrap was originally presented, includes some suggestions that attenuate the sample impoverishment problems. For fixed parameter estimation a small amount of synthetic Gaussian noise is added to jitter the resampled points away from each other. An interpretation of this approach is that a Gaussian kernel estimate is used to smooth the particle representation of the posterior. The width of the kernel can be chosen using standard kernel smoothing rules. To overcome the problem induced by outlier measurements, Gordon et al. [78] suggests to introduce a prior editing in the sample prediction step. The editing procedure decides only to include the updated sample in the set if its likelihood is large enough. The algorithm keeps generating samples until N candidates have been accepted. The risk of filter divergence is reduced since the samples are focused on an area of high future likelihood. Since a future measurement is needed in order to evaluate this likelihood, this technique will delay the state estimate calculations for one time step. Carpenter et al. [37, 38] suggests using stratified sampling as a mean to overcome some of the problems with the original algorithm. Instead of sampling one element from each term in the mixture density (6.23), a mixture density of the posterior is formed and a random number of samples is generated from each term of this mixture. The random integer number of draws from each term is chosen such that it is on the average proportional to the normalized weight of this term in the mixture and such that the total number of samples equals N. An efficient algorithm for this stratified sampling technique is given in [37, 38].

Sequential Importance Sampling

The Bayesian importance sampling procedure of Section 6.2.2 can straightforwardly be extended to recursive estimation. In general, the integrals of the Bayesian recursion (6.21)–(6.22) all have the form

$$I(x_{t+1}) = \int_{\mathbb{R}^{nt}} f(x_{t+1}, x_t) p(\mathbb{X}_t \,|\, \mathbb{Y}_t)\, d\mathbb{X}_t.$$

An importance sampling approximation to this integral is obtained by generating samples from a proposal $q(\mathbb{X}_t \,|\, \mathbb{Y}_t)$ and estimating the integral with a weighted Monte Carlo sum. By choosing a recursively updated proposal distribution

$$q(\mathbb{X}_t \,|\, \mathbb{Y}_t) = q(x_t \,|\, \mathbb{X}_{t-1}, \mathbb{Y}_t) q(\mathbb{X}_{t-1} \,|\, \mathbb{Y}_{t-1}) \tag{6.24}$$

it is possible to obtain a recursion for the cloud of particles $\{x_t^i\}_{i=1}^N$ and the corresponding set of importance weights $\{w_t^i\}_{i=1}^N$ that can be used for Monte Carlo approximation with respect to the marginal filtering density

$$p(x_t \,|\, \mathbb{Y}_t). \tag{6.25}$$

Both Doucet [60] and Liu and Chen [110] develop general frameworks based on importance sampling with respect the complete history of the state $\mathbb{X}_t$. Practically one often refrains from this for computational reasons. Moreover, the main interest

often lies in estimates given by integrals with respect to (6.25) or some equivalent prediction density. Thus, we trade some generality for notational conveniency and practical applicability and study the propagation of the marginal distribution (6.25) instead of the complete history of the states, as given in [60, 110].

The recursion is initiated by assuming that a cloud of particles representative for the filter density $p(x_{t-1} \mid \mathbb{Y}_{t-1})$ is given. In the Bayesian importance sampling framework this cloud of particles is supposed to be an i.i.d. draw from the proposal $q(x_{t-1} \mid \mathbb{Y}_{t-1})$, yielding the weighted approximation

$$p(x_{t-1} \mid \mathbb{Y}_{t-1}) \approx \sum_{i=1}^{N} w_{t-1}^i \delta(x_{t-1} - x_{t-1}^i), \tag{6.26}$$

where w_{t-1}^i are the normalized importance weights

$$w_{t-1}^i \propto \frac{p(x_{t-1}^i \mid \mathbb{Y}_{t-1})}{q(x_{t-1}^i \mid \mathbb{Y}_{t-1})}.$$

One key feature with the importance sampling procedure is that the proportionality constant of the desired density does not necessarily have to be known in order to compute the importance weights. This makes the importance sampling method extra interesting in Bayesian frameworks. From the Bayesian recursion (6.22) the desired unnormalized density is conceptually given by

$$p(x_t \mid \mathbb{Y}_t) \propto p(y_t \mid x_t) \int_{\mathbb{R}^n} p(x_t \mid x_{t-1}) p(x_{t-1} \mid \mathbb{Y}_t)\, dx_{t-1}.$$

Inserting the estimate (6.26) yields that approximately,

$$p(x_t \mid \mathbb{Y}_t) \propto p(y_t \mid x_t) \sum_{i=1}^{N} p(x_t \mid x_{t-1}^i) w_{t-1}^i$$

which can be regarded as a mixture density with N terms. An importance weighted approximation to this density is obtained by drawing one proposal sample for each term in this mixture sum and calculating an appropriately chosen importance weight for this sample. Hence, given (6.26) we generate

$$x_t^i \sim q(x_t \mid x_{t-1}^i, \mathbb{Y}_t) \qquad i = 1, \ldots, N$$

from a suitable proposal $q(x_t \mid x_{t-1}, \mathbb{Y}_t)$ that may be conditionally dependent on the particle x_{t-1}^i used in term i. The resulting estimate of the filter density is

$$p(x_t \mid \mathbb{Y}_t) \approx \sum_{i=1}^{N} w_t^i \delta(x_t - x_t^i),$$

where the normalized importance weight for each term is given by

$$w_t^i \propto w_{t-1}^i \frac{p(y_t \mid x_t^i) p(x_t^i \mid x_{t-1}^i)}{q(x_t^i \mid x_{t-1}^i, \mathbb{Y}_t)}. \tag{6.27}$$

The Sequential Importance Sampling (SIS) algorithm thus consists of recursively generating N sample paths using the proposal $q(x_{t+1} \mid x_t, \mathbb{Y}_t)$ and propagating the importance weights according to (6.27).

A generic weight used in the sequential importance sampling algorithm is a function of the collected measurements $\mathbb{Y}_t$, and hence can be regarded as a random variable if the measurement are given a stochastic interpretation. The variance of this random variable is a measure of the efficiency of the importance sampling algorithm. Optimally, the posterior distribution should be used as importance sampling distribution at each iteration. If this was possible, all weights would be equal to one and hence the variance of the weights would be zero. However, with an importance function on the form (6.24), the variance of the importance weights can only increase with time [60, 101]. This degeneracy of the algorithm practically appears after a few iterations, and will result in a particle cloud where only one candidate state has a non-negligible weight. This effect is suppressed by inserting a resampling step using the SIR procedure whenever a significant degeneracy of the particle weights is observed. The resampling step eliminates the trajectories that have very small normalized importance weights and multiplies the trajectories with high normalized weights.

The effective sample size is a measure for determining the degeneracy of the particle cloud introduced by Kong et al. [101]. The relative efficiency of the importance sampling procedure is reflected by the ratio between the variance of the importance sampling estimate, and the variance of the estimate f_N^{opt} obtained if perfect Monte Carlo simulation had been possible. In the sequential framework, the importance sampling estimate is computed using $N \gg 1$ i.i.d. samples from $q(x_t \mid \mathbb{Y}_t)$,

$$f_N = \frac{\sum_{i=1}^N w_t(x_t^i) f(x_t^i)}{\sum_{i=1}^N w_t(x_t^i)}, \qquad \text{where} \qquad w_t(x_t^i) \propto \frac{p(x_t^i \mid \mathbb{Y}_t)}{q(x_t^i \mid \mathbb{Y}_t)},$$

while the perfect Monte Carlo estimate is formed using $N \gg 1$ i.i.d. samples from $p(x_t \mid \mathbb{Y}_t)$,

$$f_N^{\text{opt}} = \frac{1}{N} \sum_{i=1}^N f(x_t^i).$$

For smooth functions $f(\cdot)$ Kong et al. [101] show that

$$\frac{\mathsf{Cov}_{q(\cdot|\mathbb{Y}_t)}\left(f_N\right)}{\mathsf{Cov}_{p(\cdot|\mathbb{Y}_t)}\left(f_N^{\text{opt}}\right)} \approx 1 + \mathsf{Cov}_{q(\cdot|\mathbb{Y}_t)}\left(w_t(x_t)\right),$$

where

$$w_t(x_t^i) = \frac{p(x_t^i \mid \mathbb{Y}_t)}{q(x_t^i \mid \mathbb{Y}_t)} \qquad \text{and} \qquad x_t^i \sim q(\cdot \mid \mathbb{Y}_t).$$

The effective sample size is defined by

$$N_{\text{eff}} \triangleq \frac{N}{1 + \mathsf{Cov}_{q(\cdot|\mathbb{Y}_t)}\left(w_t(x_t^i)\right)} = \frac{N}{\mathsf{E}_{q(\cdot|\mathbb{Y}_t)}\left(w_t(x_t^i)^2\right)}.$$

It determines the degeneracy of the importance sampling procedure. This quantity is impossible to evaluate analytically, but may be estimated by

$$\hat{N}_{\text{eff}} = \frac{1}{\sum_{i=1}^{N} {w_t^i}^2} \tag{6.28}$$

where w_t^i are the normalized importance weights of the particle cloud. When $\hat{N}_{\text{eff}}$ falls below some user defined threshold N_{thres}, the SIR resampling procedure is applied to the set $\{x_t^i\}_{i=1}^N$. The Sequential Importance Sampling (SIS) procedure with resampling test, using the effective sample size, is summarized in the algorithm below.

Algorithm 6.6 (SIS with Resampling)

1. Set $t = 0$, $w_{-1}^i = \frac{1}{N}$ and generate N samples $\{x_0^i\}_{i=1}^N$ from $q(x_0 \,|\, y_0)$.
2. Update the weights

$$w_t^i = w_{t-1}^i \frac{p(y_t \,|\, x_t^i) p(x_t^i \,|\, x_{t-1}^i)}{q(x_t^i \,|\, x_{t-1}^i, \mathbb{Y}_t)} \qquad i = 1, \ldots, N.$$

3. Normalize the weights $w_t^i := \gamma^{-1} w_t^i$, where $\gamma = \sum_{j=1}^N w_t^j$.
4. If $\hat{N}_{\text{eff}} \geq N_{\text{thres}}$ jump to item 6.
5. Generate a new set $\{x_t^{i\star}\}_{i=1}^N$ by resampling with replacement N times from the discrete set $\{x_t^j\}_{j=1}^N$ where $\Pr(x_t^{i\star} = x_t^j) = w_j$. Reset the weights $w_t^i = \frac{1}{N}$.
6. Predict the resampled states $x_{t+1}^i \sim q(x_{t+1} \,|\, x_t^{i(\star)}, \mathbb{Y}_{t+1})$, increase t and iterate to item 2.

The framework of sequential importance sampling is very general. Many procedures proposed in the literature are generalizations or developments based on the simple algorithm above. In fact, the major differences between this algorithm and Algorithm 6.5 is that in Algorithm 6.6, the importance weights are calculated sequentially and resampling is only performed when actually needed. However, the general choice of importance distribution yields greater possibilities to affect the character of the algorithm. By applying importance sampling to trajectories of state sequences the procedure can also be more general than presented in Algorithm 6.6. The sequential importance sampling method described in Algorithm 6.6 is in general computationally less demanding than Algorithm 6.5. This may enable the use of larger N and thus better Monte Carlo approximation.

Other recursive Monte Carlo approaches

The recursive SIR and the sequential importance sampling filters are the by far most common particle filters for recursive estimation. A few alternative Monte Carlo approaches are summarized below.

Bølviken et al. [30] argue in favor of the rejection sampling method over the importance sampling techniques described above. Their main argument is that with rejection sampling, a draw exactly from the posterior density is obtained, and no weighting of the candidate set is needed. But with the rejection sampling procedure it is impossible to predict the number of iterations of the algorithm for obtaining a sample of fixed size N. Hence, it is not advisable to use rejection sampling in on-line implementations.

Tanizaki [143] propose a non-recursive Monte Carlo filter and compares it with a numerical integration method and an importance sampling algorithm. The evaluation is performed using low dimensional linear and non-linear state space estimation benchmark problems. The conclusion drawn by Tanizaki is that the proposed procedure does not work well when the range of the state variables is unrestricted. Since the method is non-recursive it is not interesting for on-line applications.

The branching and interacting particle filters of Crisan et al. [41], and Del Moral [49] are similar to the algorithms presented in this subsection. The interacting particle filters have a random number of particles that represent the posterior at each iteration. The interacting particle filters coincide with the Bayesian bootstrap and the sequential importance sampling methods from a practical viewpoint, but are more restricted in the choice of proposal distribution. Greater emphasis is instead put on theoretical considerations within this framework. Both Crisan et al. [41], and Del Moral [49] present several convergence results and they utilize a rather general measure theoretic presentation. The framework is also applied to continuous time filtering. Some aspects on regularization of the particle methods for continuous time filtering are given by Le Gland et al. [108]. The regularization is introduced to overcome problems with divergence of the particle cloud in prior importance sampling.

6.4.2 Implementational Issues

Resampling

Resampling is required every iteration of the Bayesian Bootstrap filter, and whenever $\widehat{N}_{\text{eff}} < N_{\text{thres}}$ in the sequential importance sampling procedure. A direct implementation of this resampling would consist of generating N i.i.d. variables from $\mathcal{U}(0,1)$, sort these in ascending order and compare them with the cumulative sum of the normalized weights. The best sorting algorithms has a complexity of $O(N \log N)$, and it has been noted in practical applications that this severely limits the range of how far N can be increased.

Several approaches to suppress the complexity of the resampling step has been suggested. Beadle and Djurić [10] decreases the computational burden by sampling every candidate with equal probability but inserting several copies of the resampled

candidate each time it is chosen. The number of copies is proportional to the weight of the chosen candidate. The authors do issue a warning that samples with very small weight will never be resampled at all using their method. However, as noted by Doucet [60], the following classical algorithm for sampling N *ordered* i.i.d. variables $\{u_i\}_{i=1}^N$ from $\mathcal{U}(0,1)$ can be used to implement exactly the resampling in $O(N)$ time.

Algorithm 6.7 (Sampling of ordered $\mathcal{U}(0,1)$ ***variables [126, p. 96])***

1. Sample $\tilde{u}_i \sim \mathcal{U}(0,1)$, for $i = 1, \ldots, N$.
2. Set $u_N = \sqrt[N]{\tilde{u}_N}$.
3. Compute $u_i = u_{i+1}\sqrt[i]{\tilde{u}_i}$, for $i = N-1, \ldots, 1$.

In Matlab syntax this algorithm simply becomes

```
>> u = fliplr(cumprod(rand(1,N).^(1./(N:-1:1))));
```

which obviously is an $O(N)$-calculation. With an ordered i.i.d. sample from $\mathcal{U}(0,1)$ at hand, the resampling is straightforwardly implemented by comparing this sample with the cumulative sum of the normalized weights.

Optimal Importance Function

The choice of importance distribution will generally determine the performance of the algorithm and particularly the speed of degeneracy of the normalized weights. One way to choose a suitable $q(x_t \mid x_{t-1}, \mathbb{Y}_t)$ is to select the function that minimizes the degeneracy introduced in each iteration of the algorithm. The importance sampling distribution should thus minimize the variance of the importance weights $\{w_t^i\}_{i=1}^N$, given the set of candidates at the previous step of the algorithm $\{x_{t-1}^i\}_{i=1}^N$, and the measurements $\mathbb{Y}_t$. Doucet [60] shows that the optimal choice for this criterion is given by

$$q(x_t \mid x_{t-1}^i, \mathbb{Y}_t) = p(x_t \mid x_{t-1}^i, y_t).$$

With this choice of optimal importance function the importance weight recursion becomes

$$w_t^i \propto w_{t-1}^i \frac{p(y_t \mid x_t^i) p(x_t^i \mid x_{t-1}^i)}{p(x_t^i \mid x_{t-1}^i, y_t)} = w_{t-1}^i \, p(y_t \mid x_{t-1}^i).$$

In order to use the optimal importance function, one has to be able to sample from $p(x_t \mid x_{t-1}^i, y_t)$ and to evaluate

$$p(y_t \mid x_{t-1}^i) = \int p(y_t \mid x_t) p(x_t \mid x_{t-1}^i) \, dx_t,$$

at least up to a normalization constant. In the general case this is impossible, but for problems that can be written on the form

$$\begin{aligned} x_{t+1} &= f_t(x_t) + w_t \\ y_t &= H_t x_t + e_t \end{aligned} \tag{6.29}$$

with Gaussian independent noises, the optimal importance function is straightforwardly derived, see [60]. Doucet [60] suggests to utilize local linearization techniques when the problem is not on the form (6.29). We conclude the discussion on optimal importance functions with an example of such techniques.

Example 6.3
Consider the bearings-only tracking problem from Example 3.2. The state space model for this recursive estimation problem is

$$\begin{aligned} x_{t+1}^{(1)} &= x_t^{(1)} + w_t^{(1)} \\ x_{t+1}^{(2)} &= x_t^{(2)} + w_t^{(2)} \\ y_t &= \tan^{-1}\left(x_t^{(2)}/x_t^{(1)}\right) + e_t. \end{aligned}$$

This model is linear in the state update and nonlinear in the measurement relation. A change to polar coordinates gives a nonlinear state update and a linear measurement relation

$$\begin{aligned} r_{t+1} &= \sqrt{r_t^2 + w_t^{(1)} r_t \sin\varphi_t + w_t^{(2)} r_t \cos\varphi_t + w_t^{(1)} w_t^{(2)}} \\ \varphi_{t+1} &= \tan^{-1}\left(\frac{r_t \sin\varphi_t + w_t^{(2)}}{r_t \cos\varphi_t + w_t^{(1)}}\right) \\ y_t &= \varphi_t + e_t. \end{aligned}$$

The range state r_t does not enter the likelihood, so the optimal proposal distribution for r_t^i is the state transition equation given the candidate r_{t-1}^i and φ_{t-1}^i. The optimal proposal for the bearings state does not have a simple form, but a linearization of the coordinate transformation around r_{t-1}^i and φ_{t-1}^i yields that

$$\lambda_t^i = \mathsf{Cov}\left(\varphi_t \,|\, r_{t-1}^i, \varphi_{t-1}^i\right) \approx 200^2 \left(\left(\frac{\sin(2\varphi_{t-1}^i)}{2}\right)^2 + \left(\frac{\cos\varphi_{t-1}^i}{r_{t-1}^i}\right)^2\right),$$

where the covariance matrix of the process noise w_t from Example 3.2 has been inserted. This suggests an approximatively optimal importance density given by the Gaussian distribution

$$\varphi_t \,|\, r_{t-1}^i, \varphi_{t-1}^i, y_t \sim \mathcal{N}\left(\frac{0.1\varphi_{t-1}^i + \lambda_t^i y_t}{0.1 + \lambda_t^i}, \frac{0.1\lambda_t^i}{0.1 + \lambda_t^i}\right)$$

where the measurement noise covariance from Example 3.2 has been used. The approximatively optimal importance distribution above can also be used to compute importance weights.

Parallelizing the calculations

The weight update and particle diffusion can straightforwardly be calculated independently for each candidate state vector in the set. The resampling is however not possible to parallelize and this is also the part that is most time-consuming. Therefore it is of most importance that a good proposal is chosen to minimize the frequency of resampling.

One way to achieve a parallel implementation is to run several particle filters in parallel. A very high degree of parallelism cannot be achieved since each filter still must have a rather large sample size N, so that the particle cloud of each filter resembles the posterior and does not diverge. The state estimates are based on a Monte Carlo average over the total set of all points in all filters, with the weights appropriately normalized so that all the weights sum to unity.

6.4.3 Applications

The particle filters above have lately been applied to several recursive estimation problems. Target tracking applications are very common to benchmark new ideas for recursive estimation. Avitzour [5], Gordon [76], and the original article of Gordon et al. [78], use different versions of the recursive SIR. Other Monte Carlo algorithms for sequential target tracking in clutter is covered by Liu et al. [111], while target tracking in glint noise is considered by Gordon and Whitby [77].

In Chapter 7 the sequential algorithms above are applied to the terrain navigation application. Nonlinear recursive estimation problems arise frequently in navigation applications. Integration of inertial navigation system information with position fixes from satellite systems such as GPS have been studied extensively and mainly tackled with extended Kalman filter techniques. Recently, Carvalho et al. [39] presented simulation results using particle filters. Their conclusion is that the particle filters are greatly superior to the standard linearization techniques of extended Kalman filtering.

6.5 Summary

The simulation based methods are general numerical integration and optimization tools that are tightly coupled to the type of problems often occuring in Bayesian estimation. Integral estimation is performed by sample averaging over a large number of random samples generated from a desired distribution. The methods of importance and rejection sampling are suitable when an approximate description of the desired density is available. The suite of Markov Chain Monte Carlo methods offer solutions to problems of higher dimension, when it is harder to approximately describe the density of interest.

For recursive estimation, the numerical integration methods of Chapter 5 and the particle filters of this chapter are in several aspects very similar. Both represent the posterior density by a set of weighted points in the state space, and both update this representation with respect to the incoming measurements. The main

difference though, is that for the numerical integration methods the grid is chosen by the user, while in the Monte Carlo methods it is more or less automatically chosen by the model of the problem. The computational complexity of the standard numerical integration methods grows exponentially with the size of the problem, while the particle filters in general do not suffer from such a limitation. Moreover, the particle filters are very simple to implement compared to the numerical, grid based, methods.

7

Terrain Navigation

The terrain navigation application requires an on-line solution of a highly nonlinear recursive estimation problem. In this work, we emphasize the statistical approach to recursive estimation in general, and the Bayesian paradigm in particular. The basic concept of terrain navigation is depicted in Figure 7.1. Measurements of absolute altitude, ground clearance, along with heading and speed information from the inertial navigation system are compared to a terrain reference map with the objective of autonomously generating reliable position estimates of the aircraft in real time. Chapter 2 presented a background to the terrain navigation application, the stochastic modeling of the problem, the optimal but intractable Bayesian solution, and an approximative implementation. The point-mass implementation was further detailed in Chapter 5. A simulation study presented in Chapter 2, showed that the suboptimal Bayesian point-mass filter (PMF) yields high navigation performance in realistic simulations. The recursive estimation model for the terrain navigation problem was given in (2.4),

$$\begin{aligned} x_{t+1} &= x_t + u_t + w_t \\ y_t &= h(x_t) + e_t \end{aligned} \qquad t = 0, 1, \dots \tag{7.1}$$

accompanied by density functions for the initial position x_0, and the disturbances e_t and w_t, respectively. It is the unstructured nonlinear terrain map, entering the measurement equation of the model (7.1), that makes this problem challenging.

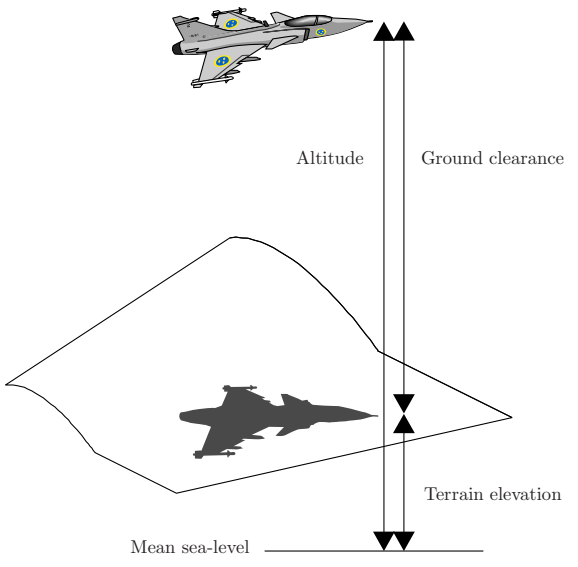


Figure 7.1: The principle of terrain navigation.

The general overview of the terrain navigation application given in Chapter 2 is complemented by some additional background material in Section 7.1. The section contains a discussion regarding the concept of integrating the position estimates from the terrain navigation system with the inertial navigation system. A detailed description of previously presented approaches to this application is also given in Section 7.1. A much more detailed description of the terrain navigation application can be found in Bergman [16]. Characteristics of the sensors, and discussions regarding methods for estimating the vertical bias error are some of the topics handled in [16]. In Section 7.2, we utilize both the parametric and the posterior Cramér-Rao bounds from Chapter 4 in a Monte Carlo evaluation of the PMF implementation. The posterior Cramér-Rao bound is also linked to the information content in the terrain map in Section 7.3. Section 7.4 presents three particle filters from Chapter 6 applied to terrain navigation. These filters are evaluated against the Cramér-Rao bounds and the PMF implementation.

7.1 Application Background

The main advantage with the principle of terrain navigation, depicted in Figure 7.1, is that it yields position estimates autonomously. This navigation principle does not have to rely on support of any satellite or terrestrial radio system broadcasting information to the aircraft. However, terrain navigation will not work over the sea, over big lakes, over very flat terrain or while flying at large ground clear-

ance distances. It does, on the other hand, perform equally well under electronic countermeasure conditions, during day or night, and in all possible weather conditions [135]. There are several ways to utilize the components of a terrain navigation system for other tasks than sole support for an inertial navigation system. Suggestions of using terrain navigation for ground obstacle collision avoidance, terrain following and mission planning are given in [86, 87].

7.1.1 Integration of Navigation Systems

In the aircraft navigation application, ensuring the safety of the passengers is a critical issue. Errors in the construction of the navigation system or physical malfunctions of sensors may cause severe damage with risk for human life. Modern aircraft navigation systems therefore consist of several separate or partly separate navigation devices, where each device produces estimates of a subset of the needed navigation parameters. The output from the individual devices are combined, usually by means of a Kalman filter, to generate the complete set of navigation parameters of the aircraft. The redundancy between the devices can also be utilized to detect and isolate malfunctions in the different systems and thereby increase the overall system integrity. A comprehensive treatment of integrity monitoring of navigation systems is given by Palmqvist [119].

Inertial navigation systems can track highly dynamical motion very well over limited time intervals. However, the open loop configuration of inertial systems yields an inevitable, slowly varying, and growing error in the system output. A terrain navigation system supporting the inertial system with frequent position updates will limit this long term divergence while retaining the autonomous feature of the navigation application. On the other hand, the terrain navigation system demands heading and velocity information for maximum performance. Thus, a symbiosis constellation between an inertial navigation and a terrain navigation system will yield an integrated system with high performance on both a long and a short time scale.

The inertial system determines the aircraft position by integrating measurements of the aircraft acceleration in real time. The integration result is transformed into a fixed coordinate system using measurements of the aircraft angular rates. Let $a(t)$ denote a vector of measured acceleration and angular rates, and $x(t)$ denote the state vector of the inertial navigation system (INS). The state vector typically consists of the aircraft position, velocity and attitude together with the gyro and accelerometer states. The dynamic integration performed by the INS is described by a nonlinear differential equation,

$$\dot{x}(t) = f_t\big(x(t)\big) + g_t\big(a(t)\big). \tag{7.2}$$

where the nonlinear functions $f_t(\cdot)$ and $g_t(\cdot)$ involve different coordinate transformations [99]. If the acceleration and gyro measurements were noiseless, the INS state would be in perfect resemblance with the true state of the aircraft, denoted by $x_0(t)$. Likewise, letting $a_0(t)$ denote the actual acceleration and angular rates

of the aircraft the true state evolves according to a similar differential equation

$$\dot{x}_0(t) = f_t\big(x_0(t)\big) + g_t\big(a_0(t)\big). \tag{7.3}$$

Denote the aircraft position estimates produced by the terrain navigation system by $y(t)$. Neglecting that these are driven by the velocity estimates from the INS, the relation between the aircraft state vector and the position "measurement" from the terrain navigation algorithm can be described as

$$y(t) = h_t\big(x_0(t)\big) + e(t). \tag{7.4}$$

Above, the nonlinear function $h_t(\cdot)$ is the coordinate transformation of the position elements in $x_0(t)$ to the navigation frame of the terrain navigation filter. The measurement noise, $e(t)$, in (7.4) is the estimation error in the position estimate of the terrain navigation filter.

Let $x_\delta(t)$ and $a_\delta(t)$ be the errors in the INS state vector and the acceleration measurements, i.e.,

$$\begin{aligned} x_0(t) &= x(t) + x_\delta(t) \\ a_0(t) &= a(t) + a_\delta(t). \end{aligned} \tag{7.5}$$

Assuming that these errors are small, we neglect the second and higher order terms in a Taylor expansion of (7.3) and (7.4) around $x(t)$ and $a(t)$. Using (7.2) and (7.5) yields that the error state satisfies a linearized dynamical model

$$\begin{aligned} \dot{x}_\delta(t) &= \frac{\partial f_t(x)}{\partial x}\Big|_{x=x(t)} x_\delta(t) + \frac{\partial g_t(a)}{\partial a}\Big|_{a=a(t)} a_\delta(t) \\ y(t) - h_t\big(x(t)\big) &= \frac{\partial h_t(x)}{\partial x}\Big|_{x=x(t)} x_\delta(t) + e(t). \end{aligned} \tag{7.6}$$

This model describes the errors dynamics in the inertial navigation system, and how the terrain navigation estimate can be used as a measurement of these errors. Modeling the acceleration error $a_\delta(t)$ by a fixed Gaussian distributed vector over a small time interval, the system (7.6) is discretized and can be used to implement an extended Kalman filter estimator for $x_\delta(t)$. Figure 7.2 shows a schematic block diagram of the integration of terrain navigation and inertial navigation, and the system output is labeled $\hat{x}_0(t)$. Since the terrain navigation filter is driven by the velocity estimate of the INS, the estimated error in this velocity is fed back from the Kalman estimator to the terrain navigation filter. In Figure 7.2, the notation $\hat{v}_\delta(t)$ has been adopted for the estimated error in the velocity used in the terrain navigation algorithm, and the abbreviation TNF stands for terrain navigation filter. Note that in the linearization scheme in the extended Kalman filter, the system matrices should be evaluated at $x(t)$ and $a(t)$ and thus the Kalman filter actually depends on the INS state vector. This has not been depicted in Figure 7.2.

The integration, using terrain information to estimate the INS errors, demands a quantization of the estimation error covariance, utilized as measurement error covariance in the Kalman filter. Since this estimation error will depend on the

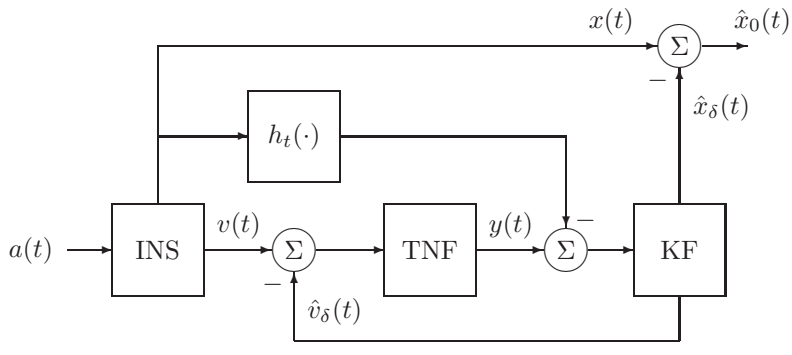


Figure 7.2: Integration of terrain information and INS.

terrain variations, the terrain navigation algorithm must calculate this estimation error on-line. Hence, the terrain navigation filter needs to output an estimate of the covariance of its estimation error,

$$R(t) = \mathsf{E}\big\{(y_0(t) - y(t))(y_0(t) - y(t))^T\big\},$$

which should be used as the measurement error covariance in the Kalman filter.

Problems of instability might occur in the constellation of Figure 7.2 due to the feedback from the Kalman filter to the terrain navigation filter. A terrain navigation algorithm leaving bad position estimates while estimating a small error covariance $R(t)$ will result in an incorrect velocity correction $\hat{v}_\delta(t)$ from the Kalman filter. Since this correction is fed back to the terrain navigation algorithm, this will probably imply an even worse estimate from the terrain navigation filter. Thus, the system in general needs to be more elaborate than indicated in Figure 7.2. The terrain navigation procedure needs to be supervised so that its position updates are used only when it is known that the filter produces reliable position estimates.

7.1.2 Approaches to Terrain Navigation

The abbreviation TAN (Terrain Aided Navigation) is commonly used for systems of the type depicted in Figure 7.2, i.e., for the concept of aiding an INS with terrain information. A brief review over some suggested approaches to terrain aided navigation was presented in Chapter 2. A survey over these methods is provided anew in this section, but here the methods are explained in greater detail and compared to the Bayesian approach.

Terrain Contour Matching

Due to limited computational resources, the initial approaches to terrain navigation relied on batch correlation between collected measurements and the digital map. The procedure of terrain contour matching (TERCOM) is such an early batch oriented techniques for navigation using terrain information.

Prior to the flight mission, several areas with high terrain variations are identified along the aircraft flight path. Reference matrices of the terrain elevation in these areas are stored in the system computer. These matrices are interpolated and aligned according to the planned vehicle orientation so that the correlation can be performed easily along the columns of the corresponding reference matrix H. Hence, the system demands that the vehicle does not manoeuvre during the data gathering, and that it has approximately correct orientation. During flight, the radar altimeter is tuned to sample the terrain elevation with a sample period adjusted so that the collected terrain elevation profile will have the same distance between consecutive samples as the distance between samples in the reference matrix H.

The most common way to perform the correlation in the TERCOM systems is to find the position in the reference matrix that minimizes the mean absolute difference (MAD) between the terrain elevation profile and the reference matrix. The MAD value is computed by calculating the absolute value of the errors along the profile and taking the mean of these absolute errors. Gathering N samples, the algorithm computes the MAD for the profile of every column m and every row displacement k in the matrix H,

$$\mathrm{MAD}_{k,m} = \frac{1}{N}\sum_{n=1}^{N} |y_n - H_{m,n+k}|.$$

The algorithm chooses the position with best correlation

$$(\hat{k}, \hat{m}) = \arg\min_{k,m} \mathrm{MAD}_{k,m}.$$

Even if the technique is old and puts severe restrictions on the manoeuvres of the aircraft, the basic TERCOM-approach is still used in several applications today. The TERCOM system is mainly used for unmanned and rarely manoeuvring vehicles such as cruise-missiles, detailed descriptions of this algorithm are provided by Baker and Clem [6], Golden [75], Siouris [135].

Sandia Inertial Terrain Aided Navigation

Sandia Inertial Terrain Aided Navigation (SITAN) is the label on a terrain navigation scheme developed at Sandia Laboratories in the late 1970s. SITAN is the earliest reported approach of a recursive solution to the terrain matching problem. The algorithm uses a modified version of an extended Kalman filter (EKF) in its original formulation by Hostetler [91]. In order to diminish the effect of terrain nonlinearities, a certain adaptive stochastic linearization technique is used in the algorithm of [91].

Several modifications of the original SITAN approach have been proposed. In order to overcome divergence problems in the filter estimates parallel EKFs have been used by Hollowell [88], and Boozer and Fellerhoff [31]. The rationale behind these extensions is that while linearizing the filters in different positions over an

area where the aircraft is supposed to be, the sensitivity to the nonlinear terrain is limited. In [31] each Kalman filter has three error states; the navigation system's easting, northing and vertical error. The idea is that as the measurements are processed, some filters will cluster around the true position and an estimate of the navigation errors can be formed. In the highly manoeuvring helicopter application considered by Hollowell [88], the filters are scalar and it is only the vertical bias that is estimated. Here, the grid of filters is fixed but translated with the movement of the helicopter, and an estimate of the position is formed by averaging over estimates from certain filters with small estimation error.

Terrain Profile Matching

The British Aerospace system Terrain Profile Matching (TERPROM) is commonly used in several military aircraft. A rather brief, and not very detailed, description of the TERPROM system is given by Davies [44]. The algorithm is a hybrid solution which in acquisition-mode correlates measurements in batch to find an initial position, and in track-mode processes measurements recursively using Kalman filtering techniques.

Viterbi Algorithm Terrain Aided Navigation

Enns and Morrell [64] propose to use a discretized version of the continuous Viterbi algorithm in the terrain navigation application. Their Viterbi Algorithm Terrain Aided Navigation (VATAN) algorithm computes approximatively a maximum a posteriori estimate of the aircraft horizontal position. The implementation uses a grid over the area of interest and translates this grid along with measurements from the inertial navigation system. The metric $L_t(x_t)$ is initiated by the prior description of the aircraft position, $L_0(x_0) = \log p(x_0)$, and calculated for each grid point according to the recursion

$$L_{t+1}(x_{t+1}) = \log p(y_{t+1} \,|\, x_{t+1}) + \max_{x_t} \left\{ \log p(x_{t+1} \,|\, x_t) + L_t(x_t) \right\}.$$

The position maximizing this metric is used as an estimate of the aircraft position

$$\hat{x}_t = \arg\max_{x_t} L_t(x_t).$$

The algorithm has not been reported used in any field tests, but in simulations described in [64] it outperforms the single EKF version of the SITAN algorithm.

Bayesian Terrain Navigation

The Bayesian solution to terrain navigation was presented in Chapter 2. The idea behind this approach is to model the recursive nonlinear inference problem in terrain navigation as accurately as possible in a statistical framework, and to compute a good approximation to the optimal Bayesian solution of this model. The main difference between this approach and the ones described previously is

that no linearization of the nonlinear terrain elevation map is needed. Instead, an approximate description of the unstructured posterior density is recursively propagated by the numerical integration method of Algorithm 5.1, the point mass filter (PMF). The PMF computes a discretized approximation to the probability density function of the aircraft position and recursively updates this discrete density with each new measurement from the radar altimeter and with the aircraft movement estimate from the inertial navigation system. The grid mesh resolution and support is controlled through three design parameters.

The main advantages of this approach were reviewed in Chapter 2. The most important issue for the terrain navigation application is the ability of the PMF to track several position hypotheses in parallel. An illustration of this ability is given in Figure 7.3 which depicts six consecutive point-mass approximations from the realistic simulations presented in Chapter 2. The point-mass approximations are

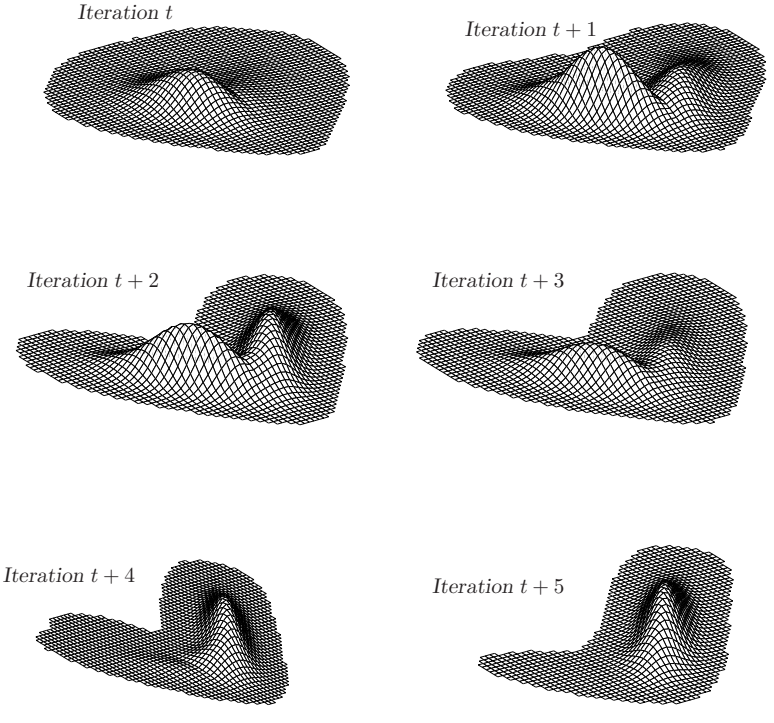


Figure 7.3: The point-mass approximation of the posterior will have several peaks if the terrain is repetitive.

extracted from a section of the simulations performed over the track in Figure 2.5 having informative, but repetitive, terrain characteristics. At iteration t the conditional density approximation has a unique maximum but a rather wide support of low, but not negligible, point mass values. Due to the underlying terrain charac-

teristic, the density splits up into two equally high peaks during the following two iterations. When the aircraft reaches an area of less repetitive terrain, a few more measurements are processed and the false peak is effectively attenuated.

The effects depicted in Figure 7.3, and the realistic simulation study described in Chapter 2 justify that the PMF solution is well suited for the terrain navigation application. The PMF has also been verified against real flight test data, reported in the Master Thesis work of Svensson [140]. According to Svensson [140], the PMF yields equally good navigation performance in the realistic simulations as on real data. It is mainly the possibility to utilize a detailed model of the noise distributions $p(e_t)$ and $p(w_t)$, and the ability of the point-mass density to accurately describe the position information, that yield the encouraging results of the PMF. Still, in a practical application it is not fully understood how to model the noise distributions $p(e_t)$ and $p(w_t)$, and how to maximize the utility of the point-mass density. In [140], the PMF is working in conjunction with a Kalman filter for estimating the vertical position error. This constellation is suggested in [16], and based on the principle of certainty equivalence.

In the following section we evaluate the average performance of the PMF using Monte Carlo simulations of the terrain navigation application. Less emphasis has here been put on the realism of the simulation conditions, and the resulting navigation error therefore only plays a minor role. Instead, we use noise distributions for which the Cramér-Rao bound for the estimation problem is easily determined and the Monte Carlo performance is compared to this fundamental bound.

7.2 Cramér-Rao Bound Evaluation

The PMF computes an approximative description of the posterior density. In this section we apply the fundamental Cramér-Rao bounds from Chapter 4 to verify that the effect of this approximation does not seriously alter the algorithm performance. We utilize Monte Carlo simulations to determine the average performance of the algorithm in two different areas, having smooth and rough terrain, respectively.

Even if the PMF is derived in a Bayesian framework, it can be evaluated in a parametric or Bayesian Monte Carlo setting. In the parametric setting, the state trajectory is fixed $\{x_i^\star\}_{i=1}^N$, and independent measurement noise is generated in M identical Monte Carlo runs. The navigation filter is applied to these data, resulting in the estimated tracks $\{\hat{x}_t^i\}_{i=1}^M$. Considering the one step ahead prediction of the aircraft position, the Monte Carlo approximation to the Cramér-Rao bound inequality was provided in Section 4.5 as

$$\sqrt{\frac{1}{M}\sum_{i=1}^{M}\left\|\hat{x}_t^i - x_t^\star\right\|^2} \gtrsim \sqrt{\operatorname{tr} P_t} \qquad t = 1, \ldots, N \tag{7.7}$$

where P_t is given by Theorem 4.3. We use a Gaussian distributed initial position with covariance P_0, and Gaussian noises w_t and e_t, with covariance Q and R, respectively. The parametric Cramér-Rao bound then follows by inserting the

model (7.1) and these noise distributions into Theorem 4.3,

$$P_{t+1} = P_t - P_t H_t (H_t^T P_t H_t + R)^{-1} H_t^T P_t + Q \qquad \text{where} \qquad H_t = \nabla_x h(x)\Big|_{x=x_t^\star}$$

is the gradient of the terrain evaluated at the true aircraft positions along the fixed track.

In the Bayesian Monte Carlo evaluation, M independent tracks are generated from (7.1) yielding M random walk realizations in two dimensions. The filter is applied to each of these tracks using measurements of the ground clearance corrupted by independently distributed measurement noise. The resulting one step ahead position estimate at time t in Monte Carlo run i is denoted $\hat{x}_t^i$. Section 4.5 yields the Cramér-Rao bound inequality

$$\sqrt{\frac{1}{M}\sum_{i=1}^{M} \|\hat{x}_t^i - x_t^i\|^2} \gtrsim \sqrt{\operatorname{tr} P_t} \qquad t = 1, \ldots, N \tag{7.8}$$

where P_t is given by Theorem 4.5. With the Gaussian noise assumption inserted into the model (7.1) we have that

$$P_{t+1}^{-1} = Q^{-1} - Q^{-1}(P_t^{-1} + Z_t + Q^{-1})^{-1} Q^{-1} \tag{7.9a}$$

where

$$Z_t = R^{-1}\,\mathsf{E}\big(\nabla_{x_t} h(x_t)(\nabla_{x_t} h(x_t))^T\big) \tag{7.9b}$$

since R is scalar and constant. The Monte Carlo estimate of this quantity is given by the sample average over the M independent tracks. The posterior bound is thus computed by averaging the terrain gradient over all the M tracks, while the parametric bound is computed from the terrain gradient at the positions given in a single track.

The minimum mean-square error, or minimum variance, estimate is the conditional mean of the posterior density. In the PMF approximation, it is given by the center of gravity of the point mass approximation. If the posterior density was perfectly described by the PMF, the minimum variance estimate would by construction satisfy the Cramér-Rao bound inequalities (7.7) and (7.8). The observed discrepancy from the Cramér-Rao bound in the Monte Carlo evaluation below can therefore be traced to the approximative point-mass density.

Table 7.1 summarizes the actual noise levels, track duration and Monte Carlo iterations used in the simulations. Two different parts of the terrain map were used in these simulations. These terrain areas are shown in Figure 7.4. The figure also depicts 200 of the tracks used in both the smooth and rough posterior Cramér-Rao bound evaluations. One of these tracks was extracted for the respective parametric Cramér-Rao bound evaluation, this track is highlighted in Figure 7.4. The PMF performance naturally depends on the parameters that affect the grid resolution and update. The chosen settings are summarized in Table 7.2. The grid truncation

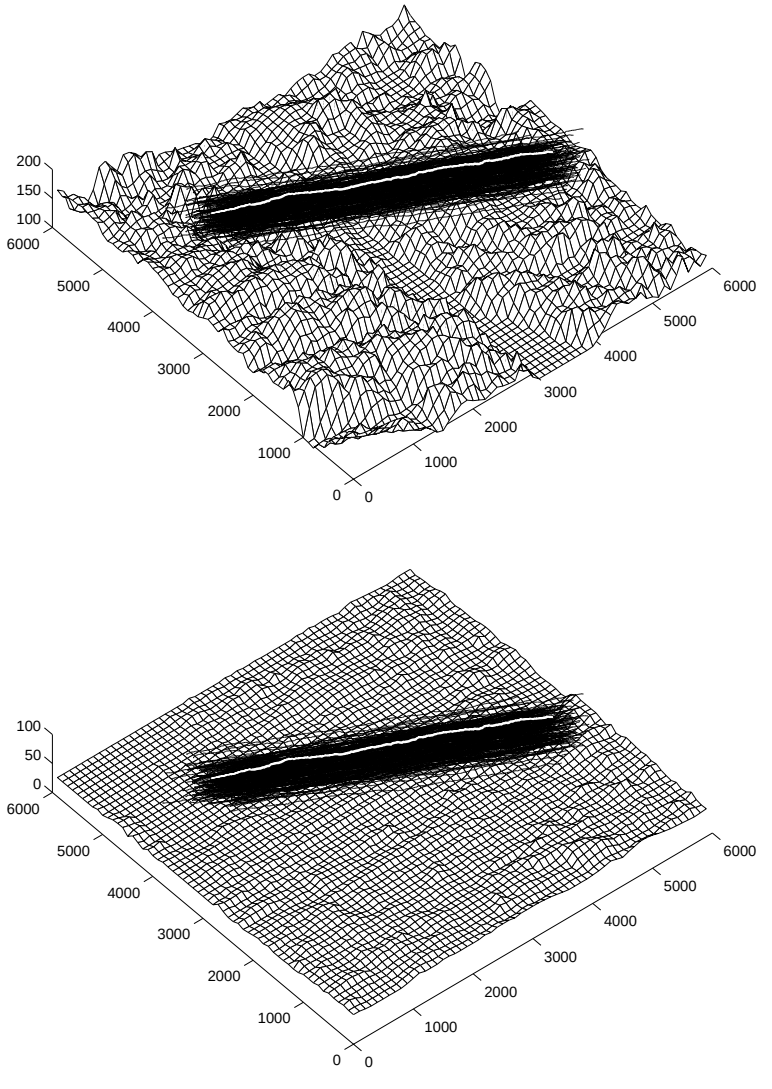


Figure 7.4: The rough and smooth simulation areas, the axes are labeled in meter scale. The first 200 simulation tracks used in the posterior simulations are shown over each map, the track used for the parametric evaluation is highlighted in white.

Initial covariance	$P_0 = 200^2 I_2$ m^2
State noise covariance	$Q = 5^2 I_2$ m^2
Measurement noise covariance	$R = 4$ m^2
Track length	300 samples
Monte Carlo runs	1000

Table 7.1: The simulation settings.

Initial grid resolution	$\delta = 50$ m
Resampling limits	$N_0 = 1000$
	$N_1 = 5000$
Truncation parameter	$\varepsilon = 0.001$

Table 7.2: Filter parameters.

and resampling parameters are identical to the ones used in the realistic simulation evaluation presented in Chapter 2.

The simulation results in the parametric Monte Carlo evaluations are depicted in Figure 7.5. Initially, a sparse grid with 50 m space between each grid point is

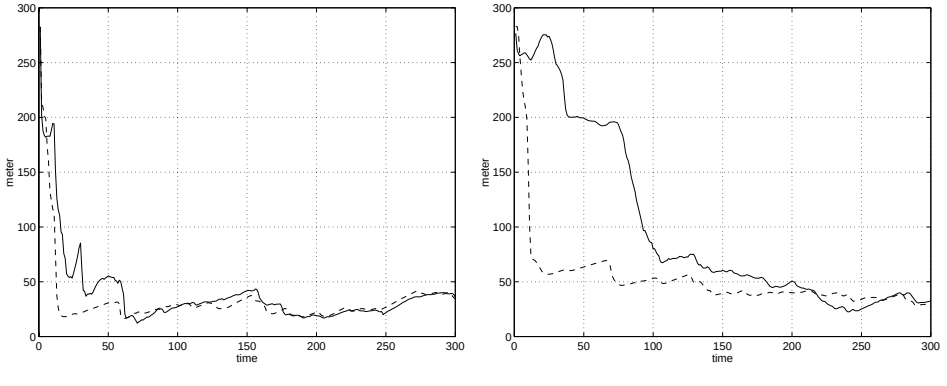


Figure 7.5: Monte Carlo root mean square error (solid) compared to the parametric Cramér-Rao bound (dashed) for the rough (left) and smooth (right) simulation areas.

used in the point-mass approximation. The grid resolution is successively increased by the PMF grid update when more information becomes available, and, after convergence, the grid resolution keeps oscillating between 6.25 m and 3.125 m. Over the rough terrain area, each measurement yields valuable information about the aircraft position and the PMF rapidly converges to a stationary error level. After

the convergence, remains in the vicinity of the bound for the remaining simulation time. Over the smooth terrain area, the less informative measurements yield a less rapid increase of the grid resolution. The sparse initial mesh, with 50 m between each grid point, clearly yields suboptimal performance during the first third of the simulation track. However, when the grid resolution increases, the algorithm performance reaches the bound and stays close to it for the remaining simulation time. During both the rough and the smooth case, the error occasionally reaches below the bound. This effect may origin from a bias towards the fixed unknown aircraft track. The effect does not seem to disappear by increasing the number of Monte Carlo iterations.

Applying the PMF to 1000 different aircraft trajectories, the Monte Carlo RMS error and corresponding posterior Cramér-Rao bound are shown in Figure 7.6. Since both the bound and the Monte Carlo RMS error are averaged over the ter-

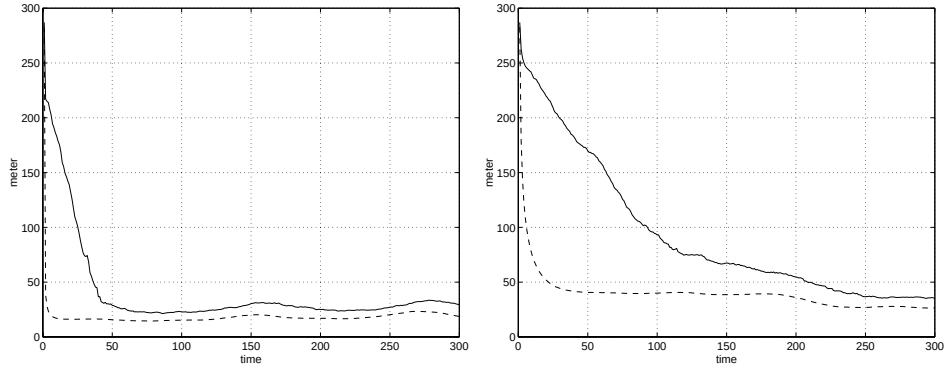


Figure 7.6: Monte Carlo root mean square error (solid) compared to the posterior Cramér-Rao bound (dashed) for the rough (left) and smooth (right) simulation areas.

rain area in this case, the plots show a smoother behavior than the parametric counterpart. In all other aspects, the results shown in Figure 7.6 indicate a similar pattern to the parametric case. The rough terrain case gives a fast convergence and close to optimal mean square error, while the smooth terrain area yields a slower convergence but equally good steady state performance. The error does not reach below the posterior bound in either the rough or the smooth case. Since the posterior Cramér-Rao bound does not assume an unbiased estimator, this indicates that the explanation of the super-effectiveness in the parametric case is sound.

The Cramér-Rao bound has been applied to the evaluation of a point-mass approximation to the Bayesian approach to terrain navigation. In several exhaustive Monte Carlo simulations over different types of terrain, the point-mass filter is shown to meet the bound as the approximation grid becomes dense. The filter performance is suboptimal during the settling phase, but after the initial convergence, the filter yields near optimal performance. In the parametric evaluation, it

occasionally happens that the mean square estimation error falls slightly below the Cramér-Rao bound. This probably origins from a bias-variance tradeoff since such behavior is not observed in the posterior case.

The Monte Carlo simulations presented above reveal the average performance of the approximate Bayesian inference procedure given in Algorithm 5.1. The fact that the mean square error of the PMF reaches the Cramér-Rao bound when the grid resolution becomes sufficiently dense, shows that the point-mass description succeeds in approximating the conditional filter density.

7.3 Terrain Information

The absolute performance of any terrain navigation filter will inevitably depend on the amount of information about the aircraft position revealed by the terrain elevation measurements. This fact is clearly obvious in the simulation results of the previous section. In this section, we connect the posterior Cramér-Rao bound to the scalar terrain information measure (2.1), suggested in Chapter 2.

Let $\mathbb{S}$ be an arbitrary region of size A inside the terrain elevation map. Consider the Bayesian estimation problem with an initial position, uniformly distributed over the region $\mathbb{S}$,

$$p(x) = \begin{cases} A^{-1} & \text{for } x \in \mathbb{S} \\ 0 & \text{elsewhere.} \end{cases}$$

and a single measurement from the relation

$$\boldsymbol{y} = h(\boldsymbol{x}) + \boldsymbol{e}$$

available. Let the additive scalar measurement noise be Gaussian distributed with zero mean and variance λ. The joint density of the states and the measurement is

$$p(x, y) = p(y \,|\, x)\, p(x) = \begin{cases} A^{-1}\, \mathrm{N}(y; h(x), \lambda) & \text{for } x \in \mathbb{S} \\ 0 & \text{elsewhere.} \end{cases}$$

Taking the logarithm of this density yields

$$\log p(x, y) = \begin{cases} -\frac{1}{2\lambda}(y - h(x))^2 + C & \text{for } x \in \mathbb{S} \\ -\infty & \text{elsewhere,} \end{cases}$$

where C is a constant, not depending on x. Hence, the "information matrix" for the Bayesian estimation problem outlined above becomes

$$J = \mathsf{E}(-\Delta_x^x \log p(x, y)) = \frac{1}{\lambda}\, \mathsf{E}\left(\nabla_x h(x) \nabla_x^T h(x) - (y - h(x)) \Delta_x^x h(x)\right).$$

Since the measurement noise $\boldsymbol{e}$ is assumed to have zero mean, we have that

$$J = \frac{1}{\lambda}\, \mathsf{E}\big(\nabla_x h(x) \nabla_x^T h(x)\big) = \frac{1}{\lambda A} \int_{\mathbb{S}} \nabla_x h(x) \nabla_x^T h(x)\, dx,$$

since the prior distribution of the states is uniform over the region $\mathcal{S}$. With a Monte Carlo approximation of this expression, a scalar terrain information measure is given by the expression suggested in (2.1),

$$\sqrt{\operatorname{tr} J} \approx \sqrt{\frac{1}{N}\sum_{i=1}^{N} \|\nabla_{x_i} h(x_i)\|^2},$$

where the points x_i are uniformly distributed over the area of interest. This information measure coincides with the one proposed in (2.1).

7.4 Monte Carlo Filters

The simulation based algorithms for recursive estimation presented in Chapter 6 utilize Monte Carlo integration techniques for approximative Bayesian estimation. Applied to terrain navigation, these filters will recursively generate a large number of candidate positions of the aircraft and determine likelihood weights assigned to each candidate trajectory. In this section we present extensive Monte Carlo evaluations using three different simulation based particle filters from Chapter 6. The Bayesian bootstrap, the sequential importance sampling and the linearized optimal importance sampling methods are studied. The simulation results are compared to the Cramér-Rao bound and the point-mass filter of Algorithm 5.1 from Section 5.2.

Assuming that the absolute altitude of the aircraft is known, or has been estimated with a small error, the objective of the terrain navigation procedure is to determine the aircraft position projected on the terrain map. The two dimensional model for this recursive estimation problem is,

$$\begin{aligned} x_{t+1} &= x_t + u_t + v_t \\ y_t &= h(x_t) + e_t \end{aligned} \qquad t = 0, 1 \ldots \tag{7.10}$$

For simplicity, the noises v_t and e_t are chosen to be Gaussian distributed, yielding that the posterior Cramér-Rao bound for the estimation problem is given by (7.9). The noise distributions and parameters in this evaluation are not primarily chosen to give realistic results for the navigation application. This is less important since comparisons are made relative to the Cramér-Rao bound. Here, we are primarily interested in the average relative performance of the different algorithms studied. The chosen parameters are summarized in Table 7.3. Gaussian distributed initial positions of the Monte Carlo tracks were generated with a fixed average position $\hat{x}_0$, and covariance P_0 given in Table 7.3. The resulting tracks generated from independent realizations of the system and measurement noises are depicted in Figure 7.7. The terrain map is shown as a contour plot behind the tracks. This is a part of the same commercial map used in the previously presented terrain navigation simulations. The posterior Cramér-Rao bound (7.9) was computed using these simulated tracks over the map.

Initial covariance	$P_0 = 100^2 I_2$ m^2
State noise covariance	$Q = 5^2 I_2$ m^2
Measurement noise covariance	$\lambda = 16$ m^2
Track length	150 samples
Aircraft velocity	$u_t = [25, 25]^T$ m/sample
Monte Carlo runs	100

Table 7.3: Parameters for the simulated tracks.

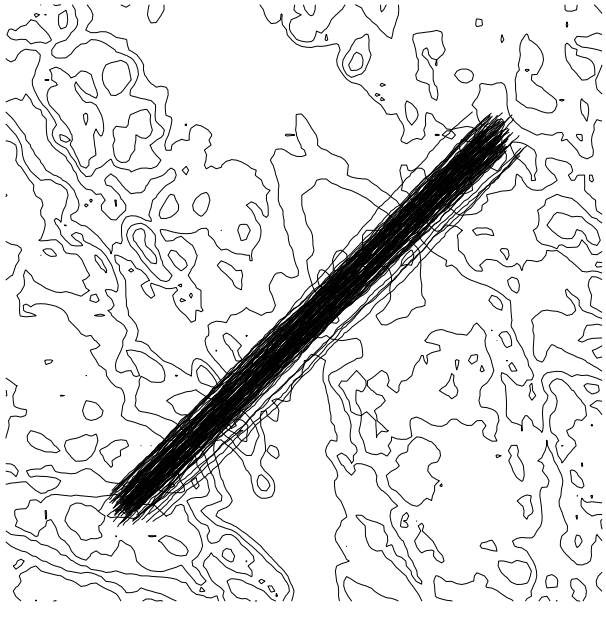


Figure 7.7: The 100 simulated aircraft tracks are depicted over a contour map of the commercial terrain map. The simulated aircraft tracks start in the lower left corner and ends in the upper right corner of the map.

The Bayesian bootstrap filter, Algorithm 6.5, performs resampling after each measurement update. Applying this algorithm to the terrain navigation model (7.10) with parameters given in Table 7.3 yields the procedure summarized below.

Algorithm 7.1 (Bayesian bootstrap)

1. Set $t = 0$, and generate N samples $\{x_0^i\}_{i=1}^N$ from $\mathcal{N}(\hat{x}_0, P_0)$.

2. Compute the normalized weights

$$w_i = \frac{\mathrm{N}(y_t; h(x_t^i), \lambda)}{\sum_{j=1}^{N} \mathrm{N}(y_t; h(x_t^j), \lambda)} \qquad i = 1, \ldots, N$$

   and determine the estimate $\hat{x}_t = \sum_{i=1}^{N} w_i x_t^i$.

3. Generate a new set $\{x_t^{i\star}\}_{i=1}^{N}$ by resampling with replacement N times from the discrete set $\{x_t^j\}_{j=1}^{N}$ where $\Pr(x_t^{i\star} = x_t^j) = w_j$.

4. Generate $v_t^i \sim \mathcal{N}(0, Q)$ for $i = 1, \ldots, N$, and predict the particle cloud

$$x_{t+1}^i = x_t^{i\star} + u_t + v_t^i \qquad i = 1, \ldots, N.$$

5. Set $t := t + 1$ and repeat at item 2.

In comparison to a Kalman filter algorithm, item 2 and 3 correspond to a measurement update step, while item 4 is the time update. Identical Monte Carlo simulations were performed over the tracks in Figure 7.7 using sample sizes N ranging from 50 to 800. Figure 7.8 shows the resulting Monte Carlo root mean square error using Algorithm 7.1. The figure also shows the fundamental posterior Cramér-Rao bound for the set of simulation tracks. The Monte Carlo root mean square error naturally decreases with increasing sample size N, although not monotonically. For the sample sizes 200 and higher, the average error after convergence is almost equal, while the speed of convergence still increases for higher sample sizes. The reason for this effect is that the aircraft position is very uncertain and the posterior filter density has a large support at the initialization of the algorithms. A large set of samples is better at approximating this posterior with wide support during the settling phase, while the algorithms using fewer samples yield a less correct description of the filter density. In the extreme case, few samples will even lead to algorithm divergence due to an incorrect description of the filter density. After half sample number 100, the filter density has become rather narrow and comparable result are obtained at almost all sample sizes. Algorithm 7.1 converges to an average error very close to the Cramér-Rao bound for all sample sizes $N > 50$.

In the sequential importance sampling algorithm, Algorithm 6.6, the degeneracy of the weights determines when to perform resampling. The algorithm described below, is the sequential importance sampling method with importance function given by the approximative prior $p(x_t \mid \mathbb{Y}_{t-1})$ from the last iteration.

Algorithm 7.2 (Importance Sampling)

1. Set $t = 0$, $w_{-1}^i = \frac{1}{N}$ and generate N samples $\{x_0^i\}_{i=1}^{N}$ from $\mathcal{N}(\hat{x}_0, P_0)$.

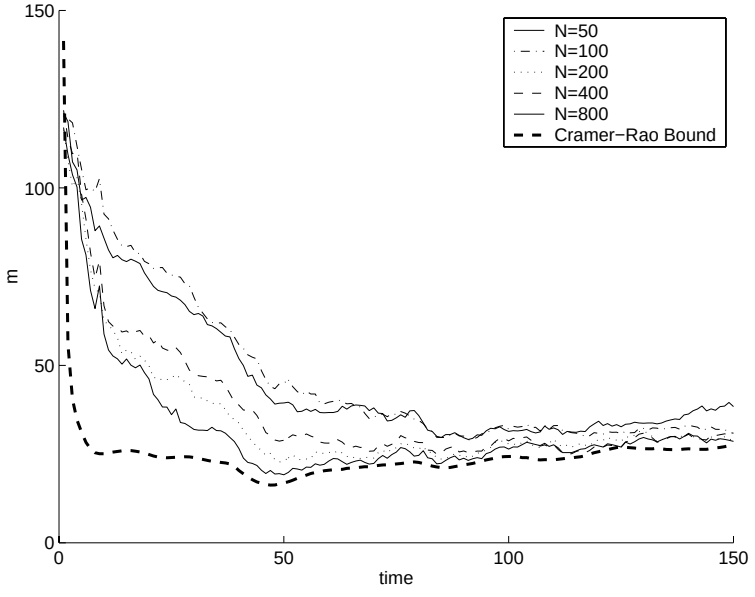


Figure 7.8: Monte Carlo RMS error and posterior Cramér-Rao bound for the Bayesian bootstrap filter, Algorithm 7.1.

2. Update the weights

$$w_t^i = w_{t-1}^i \, \mathrm{N}(y_t; h(x_t^i), \lambda) \qquad i = 1, \ldots, N$$

and normalize them

$$w_t^i := \frac{w_t^i}{\sum_{j=1}^N w_t^j} \qquad i = 1, \ldots, N.$$

Compute the estimate $\hat{x}_t = \sum_{j=1}^N w_t^i x_t^i$.

3. If $\hat{N}_{\text{eff}} < N_{\text{thres}}$ resample with replacement from the set $\{x_t^i\}_{i=1}^N$ where w_t^i is the probability of resampling the state x_t^i. Reset the weights $w_t^i = \frac{1}{N}$.
4. Generate $v_t^i \sim \mathcal{N}(0, Q)$ for $i = 1, \ldots, N$, and predict the particle cloud

$$x_{t+1}^i = x_t^i + u_t + v_t^i \qquad i = 1, \ldots, N.$$

5. Set $t := t + 1$ and repeat at item 2.

The effective sample size was defined in (6.28), as

$$\hat{N}_{\text{eff}} = \frac{1}{\sum_{i=1}^N {w_t^i}^2}$$

The threshold for the resampling operation was set to $N_{\text{thres}} = 2N/3$. The main difference between this algorithm and Algorithm 6.5, is that the weights are computed sequentially. The simulation results using this sequential importance filter are depicted in Figure 7.9. The sequential importance filter has a similar perfor-

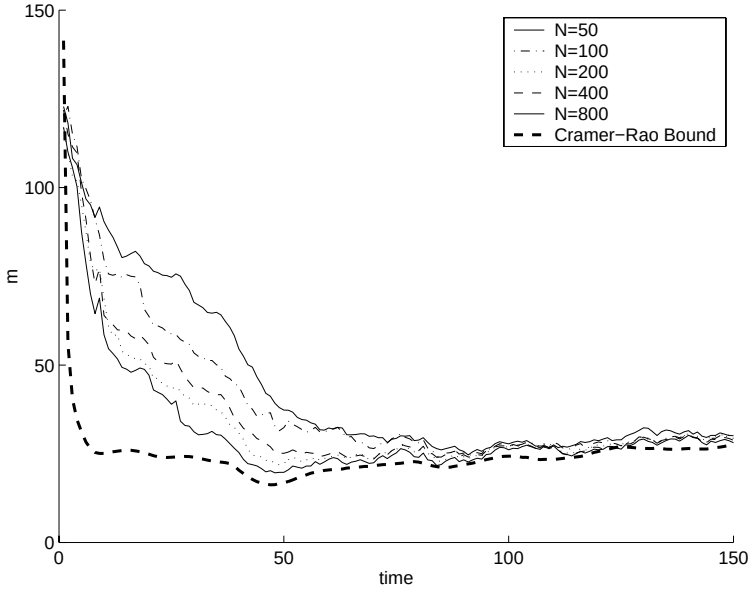


Figure 7.9: Monte Carlo RMS error and posterior Cramér-Rao bound for the sequential importance sampling filter. Prior importance function, Algorithm 7.2.

mance compared to the results of the Bayesian bootstrap presented in Figure 7.8. The filter performance increases almost monotonically with the sample size, and it is mainly the speed of convergence that is affected by the sample size parameter. However, the main difference between the algorithms is that the sequential importance sampling procedure has substantially less computational requirements for a given sample size. The resampling operation is the computationally most burdensome operation of the particle filters. Resampling of the weights needs only be performed when $\hat{N}_{\text{eff}} < N_{\text{tresh}}$ in the importance sampling algorithm due to the sequential update of the weights, while in the bootstrap filter it is a mandatory operation. In the simulations, it was observed that the resampling step is invoked on the average every fourth iteration using the importance sampling method. If the algorithms are compared on the basis of a requirement to pass a computational load limit, the results of Algorithm 7.2 for a given N should be compared to the results of Algorithm 7.1 for $N/4$. Still, the error curves of Figure 7.8 and Figure 7.9 are almost identical for the same sample size. This fact strongly speaks in favor of the importance sampling method compared to the Bayesian bootstrap. After the

settling phase, the RMS error of Algorithm 7.2 converges to a level close to the Cramér-Rao bound for all sample sizes studied.

The use of the posterior density as importance function can make the sequential importance sampling filter sensitive to outlier measurements. A better choice of importance function can also decrease the rate of the inevitable weight degeneracy. The optimal importance function, presented in Section 6.4.2, is chosen to minimize the weight degeneracy at each algorithm recursion. For the terrain navigation problem, given on the form (7.10), it is not possible to exactly sample from the optimal distribution. In the local linearization techniques promoted by Doucet [60], each sample candidate is drawn from a distribution that is a local Gaussian approximation to the true optimal distribution. In the particle prediction step, the candidate x_t^i is generated from a Gaussian distribution with covariance and mean given by

$$\begin{aligned} \Sigma_t^i &= Q - Qh_{t,i}h_{t,i}^T Q/(h_{t,i}^T Q h_{t,i} + \lambda) \\ \mu_t^i &= \Sigma_t^i(Q^{-1}(x_{t-1}^i + u_{t-1}) + h_{t,i}(y_t - h(x_t^i) + h_{t,i}^T(x_{t-1}^i + u_{t-1}))/\lambda) \end{aligned} \tag{7.11}$$

where $h_{t,i} = \nabla h(x_t^i)$, λ and Q are noise covariances from Table 7.3. The derivation is straightforward and can be found in [60]. Note though, that this technique only works with Gaussian noises e_t and v_t. Hence, the linearized optimal importance function is not applicable in the true terrain navigation application without some additional approximations since at least the measurement noise e_t is regarded as non-Gaussian. The resulting importance filter using a locally Gaussian optimal importance function follows.

Algorithm 7.3 (Optimal Importance Sampling)

1. Set $t = 0$, $w_{-1}^i = \frac{1}{N}$ and generate N samples $\{x_0^i\}_{i=1}^N$ from $\mathcal{N}(\hat{x}_0, P_0)$.
2. For each $i = 1, \ldots, N$ compute μ_t^i and Σ_t^i. Generate new samples

$$x_t^i \sim \mathcal{N}(\mu_t^i, \Sigma_t^i) \qquad i = 1, \ldots, N$$

3. Update the weights

$$w_t^i = w_{t-1}^i \frac{\mathrm{N}(y_t; h(x_t^i), \lambda)\,\mathrm{N}(x_t^i; x_{t-1}^i + u_{t-1}, Q)}{\mathrm{N}(x_t^i; \mu_t^i, \Sigma_t^i)} \qquad i = 1, \ldots, N$$

and normalize them

$$w_t^i := \frac{w_t^i}{\sum_{j=1}^N w_j} \qquad i = 1, \ldots, N.$$

Compute the estimate $\hat{x}_t = \sum_{j=1}^N w_t^i x_t^i$.

4. If $\hat{N}_{\text{eff}} < N_{\text{thres}}$ resample with replacement from the set $\{x_t^i\}_{i=1}^N$ where w_t^i is the probability of resampling the state x_t^i. Reset the weights $w_t^i = \frac{1}{N}$.

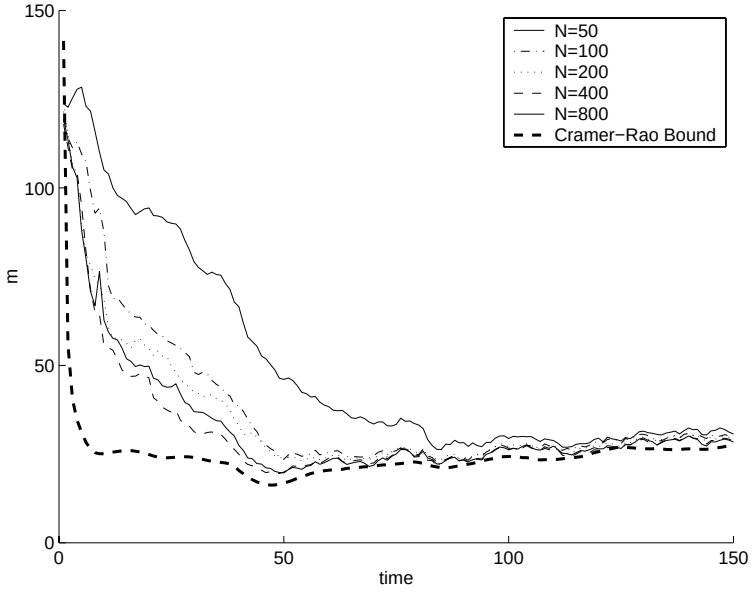


Figure 7.10: Monte Carlo RMS error and posterior Cramér-Rao bound for the sequential importance sampling filter. Locally linearized optimal importance sampling function, Algorithm 7.3.

5. Set $t := t + 1$ and repeat at item 2.

The simulation result using this filter is shown in Figure 7.10. The optimal importance function renders the same characteristics as the previously described algorithms. With increasing sample size, the filter performance increases in general, and the speed of convergence increases in particular. However, the local linearization operation (7.11) demands a substantially larger amount of computations than either of the previously described algorithms, given a fixed sample size.

During all simulations with small sample size, the algorithms occasionally lost track of the true state evolution. In the previously commented plots in Figures 7.8–7.10, the lost track runs have been manually removed and the Monte Carlo root mean square error computed using the remaining filter estimates. Table 7.4 summarizes the number of lost tracks during the evaluation. The divergence of the algorithms occurs rather seldom for the small sample sizes and not at all with a larger chosen number of particles N. Algorithm 7.2 seems to be the choice with the best robustness against sample divergence. However, all algorithms yield a comparable number of lost tracks for a given sample size and are equally robust using large N.

N	Algorithm 7.1	Algorithm 7.2	Algorithm 7.3
50	2	2	1
100	0	0	1
200	1	0	1
400	0	0	0
800	0	0	0

Table 7.4: Percentage of lost tracks during the Monte Carlo evaluations.

It is hard to draw any general conclusions about the differences in algorithm robustness based in the limited experience from this simulation evaluation. More importantly, the frequency of resampling increased substantially in the cases when the particle cloud diverged from the true state. It is therefore advisory to monitor this frequency in an on-line implementation of Algorithm 7.2 and Algorithm 7.3. Table 7.5 shows the frequency of resampling operations during the Monte Carlo evaluation after removing the simulations when the filters diverged. In the Bayesian

N	Algorithm 7.1	Algorithm 7.2	Algorithm 7.3
50	100	23	21
100	100	23	22
200	100	23	21
400	100	23	21
800	100	23	22

Table 7.5: Average resampling frequency during the Monte Carlo evaluation. The table shows the average percentage of iterations when resampling was performed.

bootstrap filter, resampling is a mandatory operation and thus Algorithm 7.1 has 100% resampling frequency. The optimal importance function generates candidate trajectories such that the sample degeneracy is minimized at each iteration. The rationale behind such a choice is to make the resampling step less frequent and thereby be able to increase the sample size. Algorithm 7.3 yields a few percent decrease in resampling frequency for the sample sizes studied, compared to Algorithm 7.2. The overhead computations required to evaluate the linearized optimal importance function (7.11) does not level this minor decrease in resampling frequency. Instead, there is a net increase in computational load when comparing Algorithm 7.2 to Algorithm 7.3. It is the very nonlinear and unstructured terrain elevation map in the terrain navigation application that diminishes the gains of this local linearization approach. The reduction in resampling frequency might be greater in problems with smoother nonlinearities, but this will naturally have to be inspected for each application. The conclusion from Table 7.5 is that Algorithm 7.2 has lowest average computational requirements due to the infrequent resampling and simple algorithm update. However, in a real time implementation one might

have to design the algorithms such that they allow for resampling at every algorithm recursion. In such a case, the importance sampling filter and the Bayesian bootstrap yield comparable performance.

The three simulation based filters were also compared to the Point-Mass Filter (PMF), given in Algorithm 5.1. A fixed grid resolution of $\delta = 12.5$ m was used in the PMF, and the truncation parameter was set to $\varepsilon = 10^{-3}$. With these parameter settings the average number of grid points used by the PMF during the 100 Monte Carlo simulations was 339. Tests with higher resolution showed no significant improvement in the PMF estimate. No lost tracks were detected using the PMF. The chosen settings for the PMF make it possible to perform a fair comparison between the results using this numerical integration method and the simulation based algorithms with sample size 400. The computational load requirements for the Bayesian bootstrap algorithm and the PMF were approximatively equal at this sample size. Figure 7.11 summarizes the resulting comparison between the simulation based algorithms of this section and the PMF algorithm of Section 5.2. All filters have comparable performance, their Monte Carlo RMS error all meet the

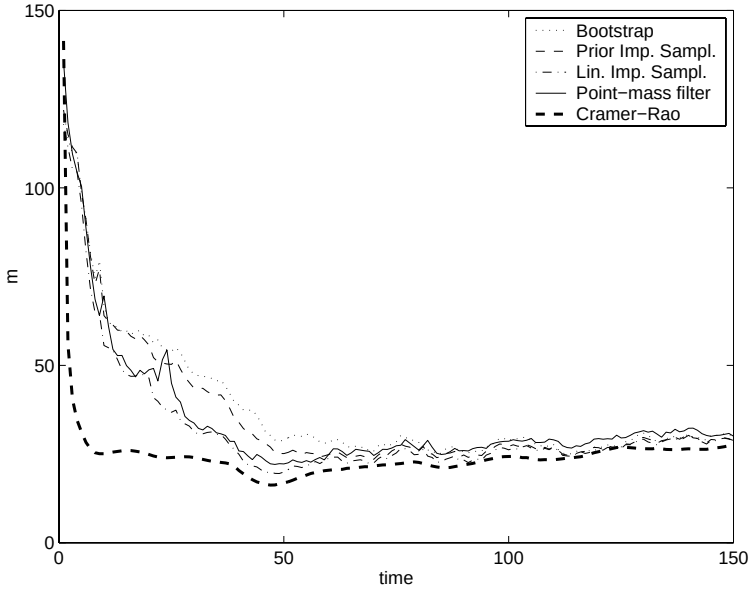


Figure 7.11: Monte Carlo RMS error and posterior Cramér-Rao bound, comparison between the three particle filters and the grid based PMF. The particle filters with sample size 400 are shown, the point-mass filter used on the average 339 grid points.

posterior Cramér-Rao bound after less than half of the simulation time. Neither the PMF nor any of the simulation based filters experienced any divergence from the true aircraft track during these simulations. The prior importance sampling method of Algorithm 7.2 gives the best performance for a given computational

requirement, while Algorithm 7.1 and the PMF demand a somewhat higher number of floating point operations. The optimal importance filter of Algorithm 7.3 puts substantially greater demands on the computational resources.

The main disadvantage with the PMF is that it is considerably more complex to implement in higher dimensions than the simulation based filters. Moreover, the growth in computational complexity makes it intractable to run the PMF in dimensions greater than three. However, since the grid in the PMF is subjectively chosen by the algorithm designer, the user has a greater potential to affect the mesh update. This may be of practical interest since, e.g., spurious outlier measurements might cause the particle cloud in the Monte Carlo algorithms to diverge as was experienced in Table 7.4.

7.5 Conclusion

The terrain navigation problem consists of intimately combining measurements from a radar altimeter, an inertial navigation system and a terrain map. This demands an unorthodox solution due to the unstructured nonlinear effects induced by the terrain map. The general Bayesian approach to this nonlinear estimation problem yields a recursive update of the complete conditional density of the aircraft position. The shape and support size of this conditional density will inevitably depend on the terrain variations.

The numerical integration point-mass approach given in Algorithm 5.1 is an approximative implementation of the Bayesian recursive solution. In extensive simulations of the terrain navigation application over different terrain types, this point-mass filter has been shown to yield approximately optimal estimation performance. It has also been shown to give encouraging results using real flight test data.

The Monte Carlo filters are alternative solutions, also generating point-mass approximations to the filter density. They show a similar near optimal performance in simulation studies compared to the Cramér-Rao bound. The Monte Carlo filters are considerably less complicated to implement than the numerical integration procedure. Particularly, they are much easier to apply in problems of higher dimension. The sequential importance sampling algorithm is the one that yields best performance at lowest computational load.

8

Target Tracking

This chapter contains illustrations of Bayesian inference in some target tracking applications. The target tracking algorithms presented in this chapter are batch oriented and therefore mainly suggested to use for off-line data analysis. Some work on recursive algorithms for target manoeuvre detection is presented in [93].

The problem of target manoeuvre detection is briefly studied in Section 8.1, while a measurement association problem is analyzed in greater detail in Section 8.2. Section 8.1 is initiated by a review of the Expectation Maximization (EM) algorithm. The EM algorithm is a procedure for computing either Bayesian MAP estimates or perform maximum likelihood estimation based on a set of incomplete, or partially unobserved, data. The technique of expectation maximization is illustrated with application to detection of target manoeuvres, at the end of Section 8.1.

The EM algorithm is also utilized for measurement-to-model association in target tracking. In Section 8.2, we consider linear Gaussian state estimation with a finite set of known linear observation models. The correspondence between the observations and the models is assumed unknown, and some of the measurements may originate from false detections, or clutter, that do not reveal anything about the true state of the system. Practically, this problem occurs in over the horizon target tracking where several layers in the ionosphere may yield more than one echo from a single target. We develop a Gibbs sampling algorithm for the joint estimation of the measurement to model association and the state sequence. The

algorithm can yield either minimum mean square error (MMSE) estimates or maximum a posteriori (MAP) estimates. This Gibbs sampler is evaluated in simulations along with two previously suggested approaches to this problem, one is based on the EM algorithm and the one other uses approximative Bayesian association. The Gibbs sampler and the EM-based algorithms are both batch oriented, while the approximative Bayesian algorithm is recursive in time.

8.1 The Expectation Maximization Algorithm

The Expectation Maximization (EM) algorithm of Dempster et al. [53] is an iterative algorithm for parameter estimation when part of the measurements are unknown. The algorithm can either be used for maximum likelihood estimation or in a Bayesian framework to obtain maximum a posteriori estimates. The parametric, maximum likelihood, formulation dates back to the work of Baum et al. [8].

Consider the maximum likelihood framework where we seek the parameter vector $\theta \in \Omega$ that maximizes the likelihood $p(y \mid \theta)$ of the observed measurement vector y. Often this function has a rather complicated structure and the maximization is hard to perform. The idea behind the EM algorithm is that there might be some hidden or unobserved data x that, if it were known, would yield a likelihood function $p(y, x \mid \theta)$ which is much easier to maximize with respect to $\theta \in \Omega$. Introduce the notation $z = [y, x]$ for the *complete data* where y are the actual measurements and x are the unobserved measurements. Since

$$p(z \mid \theta) = p(y, x \mid \theta) = p(x \mid y, \theta)\, p(y \mid \theta),$$

the log likelihood splits into two terms

$$\log p(y \mid \theta) = \log p(z \mid \theta) - \log p(x \mid y, \theta).$$

Integrating both sides with respect to a measure $f(x)$ such that $\int f(x)\, dx = 1$ will not affect the left hand side,

$$\log p(y \mid \theta) = \int \log p(z \mid \theta) f(x)\, dx - \int \log p(x \mid y, \theta) f(x)\, dx.$$

Choosing $f(x)$ as the conditional density of x given y and some candidate parameter vector θ' yields

$$\log p(y \mid \theta) = \underbrace{\mathsf{E}\left(\log p(z \mid \theta) \mid y, \theta'\right)}_{\mathcal{Q}(\theta, \theta')} - \underbrace{\mathsf{E}\left(\log p(x \mid y, \theta) \mid y, \theta'\right)}_{\mathcal{H}(\theta, \theta')}. \tag{8.1}$$

The EM algorithm only considers the first term, it is defined as alternating between forming $\mathcal{Q}(\theta, \theta')$ and maximizing it with respect to its first argument. Initializing with some θ_0, one pass of the algorithm is defined as

$$\mathcal{Q}(\theta, \theta_p) = \mathsf{E}\left(\log p(z \mid \theta) \mid y, \theta_p\right) \qquad \text{(E-step)}$$

$$\theta_{p+1} = \arg\max_{\theta} \mathcal{Q}(\theta, \theta_p) \qquad \text{(M-step)}$$

The algorithm keeps alternating between expectation and maximization until no significant improvement of $\mathcal{Q}(\theta_{p+1}, \theta_p)$ is observed in two consecutive iterations. A fundamental property of the EM algorithm is that for each pass through the algorithm the log likelihood (8.1) will monotonically increase. Below follows a brief justification of this claim, the results have been borrowed from [53].

Lemma 8.1

$$\mathcal{H}(\theta, \theta') \leq \mathcal{H}(\theta', \theta')$$

Proof Using Jensen's inequality [40] we have that

$$\begin{aligned}\mathcal{H}(\theta', \theta') - \mathcal{H}(\theta, \theta') &= \mathsf{E}\left(-\log \frac{p(x|y,\theta)}{p(x|y,\theta')} \,|\, y, \theta'\right) \geq -\log \mathsf{E}\left(\frac{p(x|y,\theta)}{p(x|y,\theta')} \,|\, y, \theta'\right) \\ &= -\log \int \frac{p(x|y,\theta)}{p(x|y,\theta')} p(x|y,\theta')\, dx = -\log \int p(x|y,\theta)\, dx = 0\end{aligned}$$

since $-\log(x)$ is convex. Equality is obtained above whenever $p(x \,|\, y, \theta') \propto p(x \,|\, y, \theta)$ where the proportionality constant must be unitary since both densities must integrate to unity. □

Theorem 8.1
For any $\theta_0 \in \Omega$

$$p(y \,|\, \theta_{p+1}) \geq p(y \,|\, \theta_p) \qquad p = 0, 1, \ldots$$

with equality if and only if both

$$\mathcal{Q}(\theta_{p+1}, \theta_p) = \mathcal{Q}(\theta_p, \theta_p)$$

and

$$p(x \,|\, y, \theta_{p+1}) = p(x \,|\, y, \theta_p)$$

Proof From (8.1) we have that

$$p(y \,|\, \theta_{p+1}) - p(y \,|\, \theta_p) = \mathcal{Q}(\theta_{p+1}, \theta_p) - \mathcal{Q}(\theta_p, \theta_p) + \mathcal{H}(\theta_p, \theta_p) - \mathcal{H}(\theta_{p+1}, \theta_p).$$

Since θ_{p+1} in the EM algorithm is the maximizing argument of $\mathcal{Q}(\theta, \theta_p)$ the difference of $\mathcal{Q}$-functions is non-negative. By Lemma 8.1 the difference of $\mathcal{H}$-functions is positive with equality if and only if $p(x \,|\, y, \theta_{p+1}) = p(x \,|\, y, \theta_p)$. □

The theorem proves that the log likelihood function will increase at each iteration of the EM algorithm. Assuming that the likelihood function is bounded for all $\theta \in \Omega$ the algorithm yields a bounded monotonically increasing sequence of likelihood values and thus it must converge to a fixed point where the conditions for equality given in Theorem 8.1 are met. Let $\theta^\star$ be a maximum likelihood (ML) estimate

of θ such that $p(y \mid \theta^\star) \geq p(y \mid \theta)$ for all $\theta \in \Omega$. Then it follows that $\theta^\star$ is a fixed point of the EM algorithm. Adding some regulatory conditions, one can also prove the converse statement, that the fixed points of the EM algorithm are in fact ML estimates, at least in a local sense [53]. One can also derive expressions for the rate of convergence of the EM algorithm, the details can be found in [53]. In general, the statement about the convergence of the EM algorithm is that it converges to a local maxima of the likelihood function $p(y \mid \theta)$.

The discussion above hold equally true under a Bayesian framework where the maximum a posterior (MAP) estimate is considered instead of the ML estimate. We simply replace the log likelihood by the posterior density

$$p(\theta \mid z) = \frac{p(z, \theta)}{p(z)} = \frac{p(z \mid \theta)p(\theta)}{p(z)}$$

and since the maximization is with respect to θ, either of

$$\mathcal{Q}(\theta, \theta_p) = \mathsf{E}\left(\log p(z \mid \theta) + \log p(\theta) \mid y, \theta_p\right)$$

or

$$\mathcal{Q}(\theta, \theta_p) = \mathsf{E}\left(\log p(z, \theta) \mid y, \theta_p\right)$$

will yield an EM algorithm having MAP estimates as fixed points.

8.1.1 A Target Tracking Example

As an illustration of the expectation maximization technique, we present an algorithm for segmentation of batch data. The observations are modeled as originating from a linear Gaussian state space model with abruptly changing system matrices. We consider detection of target manoeuvres as an application of this EM algorithm.

Consider a linear Markovian state space model for the sought n_x-dimensional target state vector x_t where the dynamics depend on the unknown binary *segmentation sequence* $\delta_t \in \{0, 1\}$,

$$\begin{aligned} x_{t+1} &= F_{t,\delta} x_t + G_{t,\delta} w_{t,\delta} \\ y_t &= H_{t,\delta} x_t + e_{t,\delta} \end{aligned} \qquad t = 1, 2, \ldots, N \tag{8.2}$$

Here, $\{w_{t,\delta}\}$ and $\{e_{t,\delta}\}$ are two independent i.i.d. sequences of zero mean Gaussian random variables with known full rank covariances $Q_{t,\delta}$ and $R_{t,\delta}$, respectively. The notation above indicates that the system parameters may have a known time dependency as well as that they depend on the current segmentation parameter δ_t.

In the manoeuvring target tracking application we consider in this work, the segmentation parameter δ_t typically affects only the process noise covariance, e.g., increasing it with a scalar fudge factor $\gamma \gg 1$ at the change instant

$$Q_{t,\delta} = (1 - \delta_t)Q_t + \gamma \delta_t Q_t. \tag{8.3}$$

The algorithm described in this work applies to general r-valued segmentations, i.e., for $\delta_t \in \{0, 1, \ldots, r\}$, and hence the model (8.2) can be seen as a mode jumping or switching model.

The initial state x_1 is Gaussian with known mean $\hat{x}_1$ and covariance P_1, the noise sequences $\{w_{t,\delta}\}$, $\{e_{t,\delta}\}$ and the initial state x_1 are mutually uncorrelated. In the data segmentation problem, a length N batch of measurement data is available for off-line processing in order to determine the corresponding length N segmentation sequence,

$$\mathbb{Y} = [y_t^T, y_2^T, \ldots, y_N^T]^T \qquad \Delta = [\delta_1, \delta_2, \ldots, \delta_N]^T.$$

We consider the Bayesian framework and seek the segmentation vector that maximizes the posterior density $p(\Delta \,|\, \mathbb{Y})$, by treating the state sequence $\{x_t\}_{t=1}^N$ as the unobserved measurement set. The prior distribution of Δ must also be specified. Explicitly, we model the prior distribution of the segmentation parameters Δ as independent Bernoulli variables,

$$\delta_t = \begin{cases} 0 & \text{with probability } q \\ 1 & \text{with probability } 1-q \end{cases} \qquad t = 1, 2, \ldots N. \tag{8.4}$$

The resulting EM algorithm for MAP estimation of the segmentation sequence is given by the iterative procedure

$$\Delta_{p+1} = \arg\max_{\Delta} \mathcal{Q}(\Delta, \Delta_p)$$

where the return function is

$$\begin{aligned} \mathcal{Q}(\Delta, \Delta_p) = \sum_{t=1}^{N} \Big(2\delta_t \log\left(\tfrac{1-q}{q}\right) - \|y_t - H_{t,\delta}\hat{x}_t^p\|^2_{R_{t,\delta}^{-1}} - \log|R_{t,\delta}| - \\ - \operatorname{tr}\left(H_{t,\delta}^T R_{t,\delta}^{-1} H_{t,\delta} P_{t,t}^p\right)\Big) + \sum_{t=1}^{N-1} \Big(- \|G_{t,\delta}^{\dagger}(\hat{x}_{t+1}^p - F_{t,\delta}\hat{x}_t^p)\|^2_{Q_{t,\delta}^{-1}} - \log|Q_{t,\delta}| - \\ - \operatorname{tr}\left(G_{t,\delta}^{\dagger T} Q_{t,\delta}^{-1} G_{t,\delta}^{\dagger} (P_{t+1,t+1}^p - 2F_{t,\delta}P_{t,t+1}^p + F_{t,\delta}P_{t,t}^p F_{t,\delta}^T)\right)\Big). \end{aligned} \tag{8.5}$$

Above, $\operatorname{tr}(A)$ is the trace of the matrix A, $|A|$ is the determinant, $\|x\|_A^2$ is the quadratic norm $x^T A x$, $A^{\dagger} = (A^T A)^{-1} A^T$ is the Moore-Penrose pseudo inverse, and

$$\begin{aligned} \hat{x}_t^p &= \mathsf{E}\,(x_t \,|\, \mathbb{Y}, \Delta_p) \\ P_{l,t}^p &= \mathsf{E}\left((x_l - \hat{x}_l^p)(x_t - \hat{x}_t^p)^T \,|\, \mathbb{Y}, \Delta_p\right). \end{aligned} \tag{8.6}$$

The quantities in (8.6) are computed by fixed interval Kalman smoothing based on the segmentation sequence Δ_p and the measurement set $\mathbb{Y}$. A derivation of the return function (8.5) is provided in Appendix 8.A.

Due to the prior (8.4), there is no dependency between δ_t for different t in the functional (8.5). Hence, the maximization step in this EM algorithm is performed for each δ_t independently. Given the estimates $\hat{x}_t^p$ and covariances $P_{l,t}^p$ the maximizing argument of (8.5) is found by comparing the return value with and without a jump at each time instant. More complicated segmentation priors would yield a maximization where there is a dependency between segmentation parameters at different times. Refer to [19] for the case of having a Poisson prior for the sequence Δ. If the segmentation sequence is Markovian, the maximization would call for dynamical programming, e.g., using the Viterbi algorithm in the maximization step. This case is thoroughly studied by Logothetis and Krishnamurthy [112].

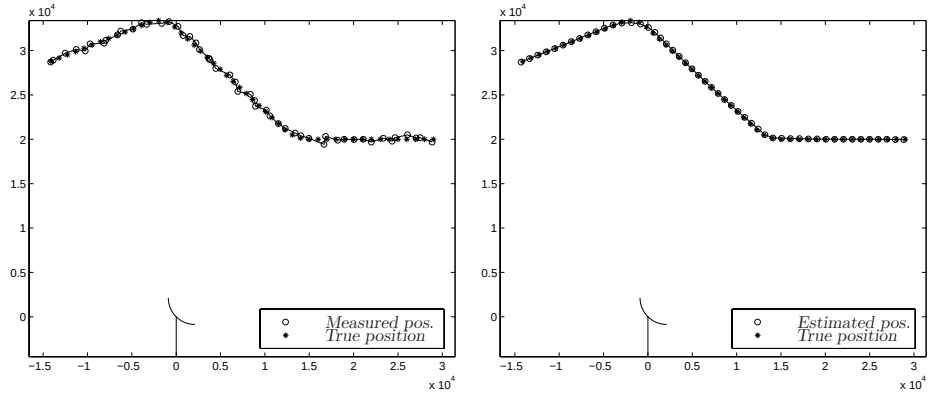
The EM algorithm for segmentation is summarized below.

Algorithm 8.1 (EM segmentation)

1. (Initialize, $p = 1$)
   Assume no jumps by setting $\Delta_1 = 0_{N\times 1}$.
2. (Expectation)
   Compute the estimates $\hat{x}_t^p$ and the error covariances $P_{t,t}^p$, and $P_{t,t+1}^p$ using fixed interval Kalman smoothing based on the segmentation sequence Δ_p.
3. (Maximization)
   Compute the next segmentation sequence Δ_{p+1} as the maximizing argument of (8.5) by choosing the alternative with highest return for each t.
4. Set $p := p + 1$ and return to item 2.

The iterations are halted when no significant improvement of (8.5) is obtained. An example of an application of the segmentation algorithm is provided below, see [22] for more details and background to this target tracking example.

Example 8.1 (Manoeuvre Detection)
A simulation example with a five state nonlinear model of an aircraft making four abrupt changes in turn rate is depicted in Figure 8.1(a). The aircraft starts to the right in the figure and flies towards the left. The position measurements are generated using a sampling period of 5 seconds and independent additive Gaussian noise with standard deviation 300 m in each channel. The target is turning during sample 15–18 and sample 35–38. Figure 8.1(b) presents the result of applying Algorithm 8.1 to the batch of data depicted in Figure 8.1(a). In the filter, a four dimensional linear model of the aircraft movement was used together with the manoeuvring model (8.3)*. The simulation model is nonlinear with aircraft turns modeled by distinct jumps in the turn rate of the model. There is a trade-off between detection sensitivity and non-manoeuvre performance since the first turn is less distinct with smaller turn rate than the second. The trade-off is controlled by choosing the three filter parameters: process noise covariance Q_t, the fudge factor γ, and the probability of not manoeuvring q. After some fine tuning of the filter*



(a) Target trajectory.

(b) Typical result using Algorithm 8.1.

Figure 8.1: Manoeuvring target trajectory and the result of applying Algorithm 8.1.

parameters the filter detected manoeuvres at sample 15 and during samples 32–40. As seen in Figure 8.1(a), the signal to noise ratio is rather high in this simulations study. Convergence is therefore obtained in only a few iterations. However, the algorithm showed a sensitivity to measurement noise and had severe problems to detect the manoeuvres in cases with smaller signal to noise ratio.

8.2 Multiple Measurement Data Association

The performance of inference under the constraint of uncertain measurement origin depends critically on the data to model association. This is, e.g., the case when tracking single or multiple targets in the presence of clutter. The nearest-neighbor Kalman filter, the Probabilistic Data Association (PDA) filter, and the optimal Bayesian filter [7] are commonly applied recursive algorithms for tracking a single target in clutter. These algorithms are applicable when there is a single model of the target measurement relation. In this section we consider an application, when several measurements from different models of the same target are detected. The problem arises in over the horizon target tracking applications where several ionospheric layers can lead to more than one resolvable echo from a single target, see Figure 8.2. We study three algorithms developed for this application. The first algorithm is a sub-optimal recursive Bayesian solution, the Multiple Simultaneous Measurement Filter (MSMF) [123]. The MSMF is based on techniques similar to the PDA filter, and is described in Section 8.2.2. The second algorithm applies the EM procedure of the last section to this tracking problem. The Expectation Maximization Data Association (EMDA) algorithm [124] treats the state sequence as the unobserved measurements and the possible association hypotheses as pa-

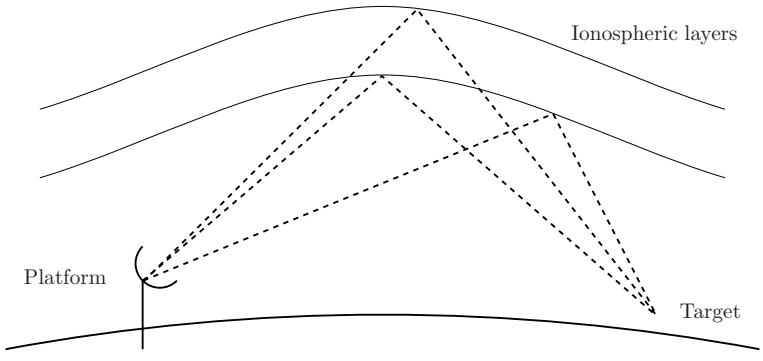


Figure 8.2: The multiple measurement data association problem arises in over the horizon target tracking.

rameters, this algorithm is detailed in Section 8.2.3. Finally, we apply the Gibbs sampler of Chapter 6 to this association problem. The derivation of this Monte Carlo algorithm is provided in Section 8.2.4. The three algorithms are compared through simulations of the tracking problem, presented in Section 8.2.5.

8.2.1 Problem Formulation

Consider a linear dynamical target state model

$$x_{t+1} = F_t x_t + G_t w_t, \quad t = 0, 1, \dots \tag{8.7}$$

where $F_t \in \mathbb{R}^{n_x \times n_x}$, and $G_t \in \mathbb{R}^{n_x \times n_u}$ are known matrices, and w_t is a zero mean white Gaussian sequence with known covariance Q_t of full rank. At time t a set of $\mu_t \geq 0$ measurements $Y_t = \{y_i(t)\}_{i=1}^{\mu_t}$ are detected, where each measurement either originate from one of n known linear measurement models or is a false detection, or clutter point. Explicitly, we let the received measurement be given by

$$y_i(t) = \begin{cases} H_1 x_t + e_1(t) & \text{for model 1} \\ H_2 x_t + e_2(t) & \text{for model 2} \\ \vdots & \vdots \\ H_n x_t + e_n(t) & \text{for model n} \\ \text{clutter} & \text{otherwise} \end{cases} \tag{8.8}$$

where $y_i(t) \in \mathbb{R}^{n_y}$ and $e_i(t)$ is a white, zero mean Gaussian sequence with known covariance R_i of full rank. The sequences $e_i(t)$, $i = 1, \dots, n$ are mutually independent and uncorrelated with the process noise w_t. The algorithms presented below admit an explicitly known time variation in all system parameters. We refrain from introducing a known time variation in the measurement models, $H_i(t)$, and

$R_i(t)$, solely to limit the notational complexity. Each measurement model in (8.8) produces a measurement with a probability P_D, which for simplicity is assumed identical for all n models. The number of clutter detections in each measurement set is Poisson distributed

$$p_c(q) = \frac{(\lambda V_S)^q}{q!} \exp(-\lambda V_S) \qquad q = 0, 1, \ldots$$

and the actual clutter measurements are uniformly distributed in the observation volume, above labeled V_S.

An explicit enumeration and description of all possible association events is needed for sake of the trailing discussion. Denote a generic *association event*, or hypothesis, at time t by the μ_t-tuple

$$\psi_t = (i_1, i_2, \ldots, i_{\mu_t}),$$

the indicator element i_j is zero if the corresponding measurement $y_j(t)$ is supposed to be due to clutter, and a non-zero value, say q, indicates that $y_j(t)$ is due to model q, where $1 \leq q \leq \mu_t$. There is no repetition of nonzero elements in ψ_t, i.e., we assume that a measurement can be due to at most one model. We also define the *target index set* by $\mathcal{T}_t = \{j \in \{1, 2, \ldots, n\} : i_j > 0\}$. To exemplify, consider the case of $\mu_t = 6$ received measurements at time t and $n = 3$ measurement models. The association event $\psi_t = (0, 0, 1, 0, 3, 0)$ would specify that measurement $y_3(t)$ originates from model number 1 and $y_5(t)$ originate from model number 3, while the remaining measurements are due to clutter. Model number 2 has failed to detect any measurements in this case and the target index set is $\mathcal{T}_t = \{3, 5\}$.

Let m_t denote the number of measurements originating from the target, i.e., the number of elements in $\mathcal{T}_t$. The number of possible association hypotheses for a given number of m_t target measurements follows by combinatorics as

$$r_t(m_t) = \frac{n!\mu_t!}{m_t!(n - m_t)!(\mu_t - m_t)!} \qquad m_t = 0, 1, \ldots, \nu_t \tag{8.9}$$

where $\nu_t \triangleq \min\{\mu_t, n\}$. The total number of association hypotheses for the measurement set received at time t is thus the sum

$$\pi_t = \sum_{m_t=0}^{\nu_t} r_t(m_t).$$

Hence, there are π_t possible association events at time t. From (8.9) it is obvious that the number of association events grows very fast with the number of received measurements. In the simulations presented in Section 8.2.5, we use $n = 3$ measurement models. Figure 8.3 shows the number of possible association hypotheses for different numbers of received measurements. Receiving five or more measurements results in more than a hundred different association events, only at that time instant.

We introduce a generic enumeration of the π_t association events, and let $\psi_t^{(i)}$ explicitly denote the ith association event. The number of target measurements, $m_t^{(i)}$, and the target index set, $\mathcal{T}_t^{(i)}$, are defined correspondingly.

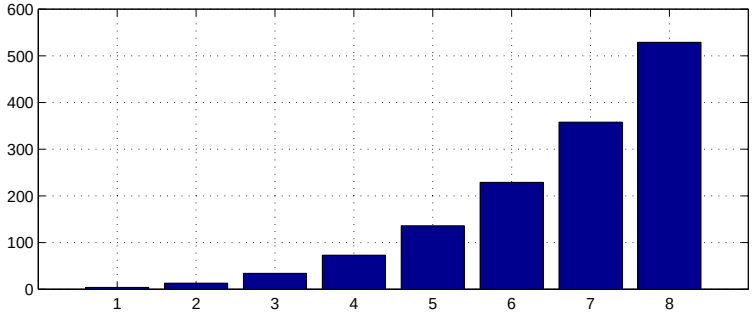


Figure 8.3: The number of possible association events π_t versus the number of received measurements for the case of $n = 3$ measurement models.

8.2.2 Multiple Simultaneous Measurement Filter

Let $\psi_t^{(i)}$ explictly denote the ith association event. The conditional filter density of the states x_t given the measurement set $\mathbb{Y}_t = \{Y_i\}_{i=0}^t$ is the mixture density

$$p(x_t \,|\, \mathbb{Y}_t) = \sum_{i=1}^{\pi_t} c_i(t) p(x_t \,|\, \psi_t^{(i)}, \mathbb{Y}_t) \tag{8.10}$$

where $c_i(t)$ is the association probability of event $\psi_t^{(i)}$,

$$c_t^{(i)} \triangleq \Pr(\psi_t^{(i)} \,|\, \mathbb{Y}_t), \quad i = 0, 1, \ldots, \pi_t. \tag{8.11}$$

Under each association hypothesis the conditional density is Gaussian, the optimal solution consists of applying a Kalman filter to each hypothesis sequence. This will lead to a mixture (8.10) where the number of terms grows exponentially with time [7]. The MSMF [123] limits this growth by replacing the Gaussian mixture in (8.10) by a single Gaussian density at each algorithm iteration t in the spirit of the Probabilistic Data Association (PDA) filter [7].

Denote the conditional mean and covariance of the state by $\hat{x}_{t|t} = \mathsf{E}(x_t \,|\, \mathbb{Y}_t)$ and $P_{t|t} = \mathrm{Cov}(x_t \,|\, \mathbb{Y}_t)$. The recursive MSMF algorithm computes approximations to this mean vector and covariance matrix, given by computing the first and second central moments of (8.10),

$$\hat{x}_{t|t} = \sum_{i=1}^{\pi_t} c_t^{(i)} \hat{x}_{t|t}^{(i)} \tag{8.12a}$$

$$P_{t|t} = \sum_{i=1}^{\pi_t} c_t^{(i)} \left(P_{t|t}^{(i)} + \hat{x}_{t|t}^{(i)} (\hat{x}_{t|t}^{(i)})^T \right) - \hat{x}_{t|t} \hat{x}_{t|t}^T. \tag{8.12b}$$

The state estimates $\hat{x}_{t|t}^{(i)}$ and error covariances $P_{t|t}^{(i)}$ are computed under the association event i at time t. Using the information form of the Kalman filter [2], they

are determined by

$$\hat{x}_{t|t}^{(i)} = \hat{x}_{t|t-1} + P_{t|t}^{(i)} \sum_{j \in \mathfrak{T}_t^{(i)}} H_{i_j}^T R_{i_j}^{-1} (y_j(t) - H_{i_j} \hat{x}_{t|t-1}) \tag{8.13a}$$

$$(P_{t|t}^{(i)})^{-1} = P_{t|t-1}^{-1} + \sum_{j \in \mathfrak{T}_t^{(i)}} H_{i_j}^T R_{i_j}^{-1} H_{i_j}. \tag{8.13b}$$

The association probabilities (8.11) are given by the prior probability of the association event and the likelihood of each Kalman filter,

$$c_t^{(i)} = \kappa^{-1} \frac{p_c(\mu_t - m_t) P_D^{m_t} (1 - P_D)^{n - m_t}}{r_t(m_t)} \prod_{j \in \mathfrak{T}_t^{(i)}} \mathrm{N}(y_j(t); H_{i_j} \hat{x}_{t|t-1}, S_j)$$

Above, κ is a normalization constant, ensuring that $\sum_{i=1}^{\pi_t} c_t^{(i)} = 1$, m_t is the number of target measurements, and

$$S_j = H_{i_j} P_{t|t-1} H_{i_j}^T + R_{i_j}.$$

The MSMF given by Pulford and Evans [123] assumes that the full rank covariance state noise enters purely additively in the estimation model, i.e., that $G_t = I$. This simplifies the algorithm since it ensures that $P_{t|t-1}^{-1}$ exists and therefore enables the use of the information form of the measurement updates in (8.13). This assumption can be removed, and the measurement update (8.13) can, e.g., be performed by sequential processing of the measurements in $\mathfrak{T}_t^{(i)}$, see [2]. The time update prediction of the estimate and error covariance is solved by a conventional Kalman step

$$\begin{aligned} \hat{x}_{t+1|t} &= F_t \hat{x}_{t|t} \\ P_{t+1|t} &= F_t P_{t|t} F_t^T + G_t Q_t G_t^T \end{aligned}$$

This completes one iteration of the recursive MSMF, a detailed derivation of this recursion is provided in [123].

8.2.3 Expectation Maximization Data Association

In the EMDA algorithm of Pulford and Logothetis [124], the complete batch of measured data $\mathbb{Y}_N = \{Y(t)\}_{t=0}^N$ is considered for off-line processing and the EM algorithm is applied to this estimation problem. The EMDA algorithm is obtained by treating the target state sequence $\mathbb{X}_N = \{x_t\}_{t=0}^N$ as the missing data and the sequence of association events $\Psi_N = \{\psi_t\}_{t=0}^N$ as the sought parameters. This yields the EM iteration

$$\widehat{\Psi}_N^{(p+1)} = \arg\max_{\Psi_N} \mathsf{E}\left(\log p(\mathbb{X}_N, \mathbb{Y}_N, \Psi_N) \,|\, \mathbb{Y}_N, \widehat{\Psi}_N^{(p)} \right) \tag{8.14}$$

where expectation is performed with respect to the unobserved data $\mathbb{X}_N$.

In [124], a target existence model is included which leads to dynamic programming via the Viterbi algorithm in the M-step of the EMDA algorithm. However, the target existence parameter introduces ambiguities into the estimator since there exists no state sequence $\mathbb{X}_N$ under the hypothesis that no target exists. Removing the target existence model leads to an M-step that can be solved independently for each t in the batch. To see this, we note that the density in (8.14) is recursively expressed as

$$p(\mathbb{X}_N, \mathbb{Y}_N, \Psi_N) = p(Y_N \,|\, x_N, \psi_N) p(x_N \,|\, x_{N-1}) \Pr(\psi_N) p(\mathbb{X}_{N-1}, \mathbb{Y}_{N-1}, \Psi_{N-1}),$$

and that terms independent of the association sequence not will affect the maximization. Hence, we can use

$$\mathcal{Q}(\Psi_N, \widehat{\Psi}_N^{(p)}) = \mathsf{E}\left(\sum_{t=1}^{N} \log p(Y_t \,|\, x_t, \psi_t) + \log \Pr(\psi_t) \,\middle|\, \mathbb{Y}_N, \widehat{\Psi}_N^{(p)} \right) \tag{8.15}$$

in liu of (8.14). This function can obviously be maximized independently for each t once the expectation has been evaluated. The maximization of (8.15) is equivalent to maximizing for each t

$$J_p(\psi_t) = -\tfrac{1}{2} \sum_{j \in \mathcal{T}_t} \| y_j(t) - H_{i_j} \hat{x}_{t|N}^{(p)} \|^2_{R_{i_j}^{-1}} + m_t \log(\rho) + \log \Pr(\psi_t) \tag{8.16}$$

where

$$\rho = \frac{V_S}{\sqrt{(2\pi)^{n_y} \left(\prod_{j \in \mathcal{T}_t} |R_{i_j}| \right)^{\frac{1}{m_t}}}}, \tag{8.17a}$$

$$\Pr(\psi_t) = \kappa^{-1} \frac{p_c (\mu_t - m_t) P_D^{m_t} (1 - P_D)^{n - m_t}}{r_t(m_t)}, \tag{8.17b}$$

and $\hat{x}_{t|N}^{(p)} = \mathsf{E}(x_t \,|\, \mathbb{Y}_N, \widehat{\Psi}_N^{(p)})$ is obtained from a fixed interval Kalman smoother under the hypothesis $\widehat{\Psi}_N^{(p)}$. In (8.17), m_t is the number of target measurements, and κ is a normalization constant ensuring that the probabilities sum to unity. Refer to [124] for a detailed derivation of these expressions.

In summary, the EMDA algorithm consists of the following steps

1. (Initialization $p = 0$)
   Choose an initial state sequence $\hat{\mathbb{X}}_N$ using the measurements $\mathbb{Y}_N$. Set $p := 1$.

2. (M-step)
   Compute the measurement to target association $\widehat{\Psi}_N^{(p)}$ maximizing the cost function (8.16) for each $t = 0, 1, \ldots, N$ using the state sequence $\hat{\mathbb{X}}_N^{(p-1)}$.

3. (E-step)
   Compute the sequence of state estimates $\hat{\mathbb{X}}_N^{(p)}$ given the current association sequence $\widehat{\Psi}_N^{(p)}$ using fixed interval Kalman smoothing.

4. Set $p := p + 1$ and return to item 2 above.

The iterations continue until no significant improvement is obtained during the maximization. The probability in (8.14) is by construction monotonically increasing with the number of passes through the EM-algorithm [8]. This ensures that a local maxima of the density is reached and that the algorithm terminates.

8.2.4 Markov Chain Monte Carlo Data Association

The Markov chain Monte Carlo algorithm presented in this section has been developed with inspiration from an application of similar ideas to estimation of parameters of jump-Markov linear systems by Doucet and Andrieu [61], Doucet et al. [62]. Like the EMDA algorithm, we consider off-line batch processing of the collected measurements $\mathbb{Y}_N$, but here we propose to perform a Gibbs sampling procedure instead.

1. Pick the initial association $\Psi_N^{(0)} = \{\psi_t^{(0)}\}_{t=0}^N$ randomly or deterministically. Set $p = 1$.

2. For each $t = 0, 1, \ldots, N$ generate a random sample from the full conditional distribution

$$\psi_t^{(p)} \sim p(\psi_t^{(p)} \mid \mathbb{Y}_N, \Psi_{\neg t}^{(p)}) \tag{8.18}$$

3. Set $p := p + 1$ and repeat at step 2.

The conditioning set in the full conditional (8.18) is

$$\Psi_{\neg t}^{(p)} = \{\Psi_{t-1}^{(p)}, \Psi_{t+1:N}^{(p-1)}\} = \{\psi_0^{(p)}, \psi_1^{(p)}, \ldots, \psi_{t-1}^{(p)}, \psi_{t+1}^{(p-1)}, \ldots, \psi_N^{(p-1)}\}.$$

Discarding an initial burn-in phase, the sample average over the simulated output from this Gibbs sampler is a minimum mean square estimate of the association sequence. With this estimate of the association sequence at hand, a Kalman smoother can be used to determine the states of the tracked system. Alternatively, Kalman smoothing can be performed between items 2 and 3, based on the sequence $\widehat{\Psi}_N^{(p)}$, and a Monte Carlo estimate of the state sequence formed from the smoothed estimates.

The full conditional distribution (8.18) is discrete with π_t possible association events at each time instant. In order to sample from this discrete distribution, we must calculate the full conditional probability for each association hypothesis conditioned on the previously drawn associations. This discrete full conditional distribution is given by

$$\begin{aligned} p(\psi_t^{(p)} \mid \mathbb{Y}_N, \Psi_{\neg t}^{(p)}) &= \frac{p(\mathbb{Y}_N \mid \Psi_{\neg t}^{(p)}, \psi_t^{(p)}) \Pr(\psi_t^{(p)})}{p(\mathbb{Y}_N \mid \Psi_{\neg t}^{(p)})} \\ &\propto p(\mathbb{Y}_N \mid \Psi_{\neg t}^{(p)}, \psi_t^{(p)}) \Pr(\psi_t^{(p)}) \end{aligned} \tag{8.19}$$

where the prior probabilities $\Pr(\psi_t^{(p)})$ are given by the model of the problem and the number of received measurements at time t. Hence, the inner loop of the Gibbs sampler relies on computing the likelihoods

$$p(\mathbb{Y}_N \mid \Psi_{\neg t}^{(p)}, \psi_t^{(p)} = \psi_t^{(i)}) \qquad i = 1, \ldots, \pi_t. \tag{8.20}$$

With these likelihood values at hand, the discrete distribution (8.19) is known up to a normalizing factor, and a sample can be generated from this distribution and inserted into $\Psi_{\neg(t+1)}^{(p)}$ at the next iteration. Following Doucet and Andrieu [61], each iteration p of the Gibbs sampler is performed by a backward filtering forward sampling procedure which has a complexity that is linear in the number of data N. We proceed with a derivation of how to compute (8.20) using this efficient backwards filtering forwards sampling scheme.

The likelihood (8.20) can be split into two parts

$$\begin{aligned} p(\mathbb{Y}_N \mid \Psi_{\neg t}^{(p)}, \psi_t^{(p)}) &= p(\mathbb{Y}_t \mid \Psi_{\neg t}^{(p)}, \psi_t^{(p)})\, p(\mathbb{Y}_{t+1:N} \mid \Psi_{\neg t}^{(p)}, \psi_t^{(p)}, \mathbb{Y}_t) \\ &= p(\mathbb{Y}_t \mid \Psi_t^{(p)})\, p(\mathbb{Y}_{t+1:N} \mid \Psi_{t+1:N}^{(p-1)}, \Psi_t^{(p)}, \mathbb{Y}_t). \end{aligned}$$

Further decomposing the first factor and writing the second factor through a marginalization yields that the likelihood (8.20) equals

$$\begin{aligned} p(\mathbb{Y}_N \mid \Psi_{\neg t}^{(p)}, \psi_t^{(p)}) = p(\mathbb{Y}_{t-1} \mid \Psi_{t-1}^{(p)})\, p(Y_t \mid \mathbb{Y}_{t-1}, \Psi_t^{(p)}) \\ \int p(\mathbb{Y}_{t+1:N}, x_t \mid \Psi_{t+1:N}^{(p-1)}, \Psi_t^{(p)}, \mathbb{Y}_t)\, dx_t. \end{aligned}$$

Inserting this expression into (8.19) we have that

$$\begin{aligned} p(\psi_t^{(p)} \mid \mathbb{Y}_N, \Psi_{\neg t}^{(p)}) \propto p(Y_t \mid \mathbb{Y}_{t-1}, \Psi_t^{(p)})\, \Pr(\psi_t^{(p)}) \\ \int p(\mathbb{Y}_{t+1:N} \mid \Psi_{t+1:N}^{(p-1)}, x_t)\, p(x_t \mid \Psi_t^{(p)}, \mathbb{Y}_t)\, dx_t \end{aligned} \tag{8.21}$$

after removing the factors that are independent of $\psi_t^{(p)}$. Given the association sequence $\Psi_t^{(p)}$, standard Kalman filter theory gives that

$$p(x_t \mid \Psi_t^{(p)}, \mathbb{Y}_t) = \mathrm{N}(x_t; \hat{x}_{t|t}, P_{t|t}) \tag{8.22}$$

while

$$p(Y_t \mid \mathbb{Y}_{t-1}, \Psi_t^{(p)}) = \mathrm{N}(Y_t; \hat{Y}_{t|t-1}, S_t)\, \frac{1}{V_S^{1-m_t}}$$

is found from the innovation of this Kalman filter, and m_t is the number of non-clutter observations under the hypothesis $\psi_t^{(p)}$.

We need an intermediate lemma in order to rewrite the last factor of (8.21).

Lemma 8.2
For any vectors Y, x, m, and matrices M, R, P of compatible dimensions

$$\int \mathrm{N}(Y; Mx, R)\ \mathrm{N}(x; m, P)\, dx = \mathrm{N}(Y; Mm, R + MPM^T) \tag{8.23}$$

Proof Let $Z \sim \mathcal{N}(0, R)$ and $x \sim \mathcal{N}(m, P)$ be independent Gaussian random vectors. Introduce $Y = Z + Mx$ with probability density function $p(Y) = \int p(Y \mid x)p(x)\, dx$, given by the left hand side of (8.23). On the other hand, Y is a linear combination of Gaussian vectors and obviously has probability density function $p(Y) = \mathrm{N}(Y; Mm, R + MPM^T)$. □

Under a specific association hypothesis sequence $\Psi_{t+1:N}$, the considered data association problem is a standard linear state space problem, which yields that

$$\mathbb{Y}_{t+1:N} = M_t x_t + U_t \tag{8.24}$$

where

$$M_t = \begin{bmatrix} \mathbb{H}_{t+1} F_t \\ \mathbb{H}_{t+2} \Phi_{t+1,t} \\ \vdots \\ \mathbb{H}_N \Phi_{N-1,t} \end{bmatrix} \quad \text{and}$$

$$U_t = \begin{bmatrix} \mathbb{H}_{t+1} G_t & 0 & \cdots\cdots & 0 \\ \mathbb{H}_{t+2} F_{t+1} G_t & \mathbb{H}_{t+2} G_{t+1} & 0 & 0 \\ \vdots & & \ddots & \vdots \\ & & & 0 \\ \mathbb{H}_N \Phi_{N-1,t+1} G_t & \cdots\cdots\cdots\cdots & & \mathbb{H}_N G_{N-1} \end{bmatrix} \begin{bmatrix} w_t \\ \vdots \\ w_{N-1} \end{bmatrix} + \begin{bmatrix} e_{t+1} \\ \vdots \\ e_N \end{bmatrix}$$

and we introduce the notation $\Phi_{t+k,t} = F_{t+k} \dots F_{t+1} F_t$ for the state transition matrix and $\mathbb{H}_t$ is the measurement matrix corresponding to generic association ψ_t. Hence,

$$\mathbb{Y}_{t+1:N} \mid \Psi_{t+1:N}^{(p-1)}, x_t \sim \mathcal{N}(M_t x_t, L_t)$$

where $L_t = \mathsf{Cov}(U_t)$ is positive definite by construction. From Lemma 8.2 we thus have that the last factor of (8.21) can be written as

$$\int p(\mathbb{Y}_{t+1:N} \mid \Psi_{t+1:N}^{(p-1)}, x_t)\, p(x_t \mid \Psi_t^{(p)}, \mathbb{Y}_t)\, dx_t = \mathrm{N}(\mathbb{Y}_{t+1:N}; M_t \hat{x}_{t|t}, L_t + M_t P_{t|t} M_t^T) \tag{8.25}$$

where $\hat{x}_{t|t}$ and $P_{t|t}$ are computed under the hypothesis $\Psi_t^{(p)}$, as noted in (8.22). In summary, the full conditional (8.21) is given by

$$p(\psi_t^{(p)} \mid \mathbb{Y}_N, \Psi_{-t}^{(p)}) \propto \mathrm{N}(Y_t; \hat{Y}_{t|t-1}, S_t)\, \Pr(\psi_t^{(p)}) V_S^{m_t} \\ \mathrm{N}(\mathbb{Y}_{t+1:N}; M_t \hat{x}_{t|t}, L_t + M_t P_{t|t} M_t^T) \tag{8.26}$$

where m_t is the number of non-clutter measurements under $\psi_t^{(p)}$, as usual.

Next, we show how the last factor of (8.26) can be expressed through a backwards-forwards recursion. The matrix inversion lemma (4.A.20) gives that

$$(L_t + M_t P_{t|t} M_t^T)^{-1} = L_t^{-1} - L_t^{-1} M_t (M_t^T L_t^{-1} M_t + P_{t|t}^{-1})^{-1} M_t^T L_t^{-1},$$

so the exponent of (8.25) can be decomposed according to

$$\begin{aligned}(\mathbb{Y}_{t+1:N} - M_t \hat{x}_{t|t})^T (L_t + M_t P_{t|t} M_t^T)^{-1} (\mathbb{Y}_{t+1:N} - M_t \hat{x}_{t|t}) = \\ \mathbb{Y}_{t+1:N}^T L_t^{-1} \mathbb{Y}_{t+1:N} - 2\hat{x}_{t|t}^T M_t^T L_t^{-1} \mathbb{Y}_{t+1:N} + \hat{x}_{t|t}^T M_t^T L_t^{-1} M_t \hat{x}_{t|t} - \\ - (\mathbb{Y}_{t+1:N} - M_t \hat{x}_{t|t})^T L_t^{-1} M_t (M_t^T L_t^{-1} M_t + P_{t|t}^{-1})^{-1} M_t^T L_t^{-1} (\mathbb{Y}_{t+1:N} - M_t \hat{x}_{t|t}).\end{aligned}$$

The first term in this expression is independent of the association event ψ_t and can thus be ignored. The determinant relation $|I + AB| = |I + BA|$ yields that

$$|L_t + M_t P_{t|t} M_t^T| = |L_t||I + P_{t|t} M_t^T L_t^{-1} M_t|,$$

and since L_t is independent of the association event $\psi_t^{(p)}$ the factor $|L_t|$ can also be ignored. Now, standard least squares estimation theory [2, 96] applied to (8.24) yields that the Fisher estimate of x_t given $\mathbb{Y}_{t+1:N}$ can be written

$$\begin{aligned}\hat{a}_{t|t+1} &\triangleq P_{t|t+1:N}^{-1} \hat{x}_{t|t+1:N} = M_t^T L_t^{-1} \mathbb{Y}_{t+1:N} \\ \Gamma_{t|t+1} &\triangleq P_{t|t+1:N}^{-1} = M_t^T L_t^{-1} M_t,\end{aligned}$$

Hence, running a backwards information filter under the hypothesis $\Psi_N^{(p-1)}$ with initial inverse covariance $P_{N|N+1}^{-1} = 0$ will render the necessary quantities for computation of the full conditional probabilities (8.26). The backwards information filter is given below, see, e.g., [80].

$$\hat{a}_{t|t} = \hat{a}_{t|t+1} + \mathbb{H}_t^T \mathbb{R}_t^{-1} Y_t \tag{8.27a}$$

$$\Gamma_{t|t} = \Gamma_{t|t+1} + \mathbb{H}_t^T \mathbb{R}_t^{-1} \mathbb{H}_t \tag{8.27b}$$

$$\Delta = \left(G_t^T \Gamma_{t|t} G_t + Q_t^{-1}\right)^{-1} \tag{8.27c}$$

$$\hat{a}_{t-1|t} = F_t^T \left(I - \Gamma_{t|t} G_t \Delta G_t^T\right) \hat{a}_{t|t} \tag{8.27d}$$

$$\Gamma_{t-1|t} = F_t^T \left(I - \Gamma_{t|t} G_t \Delta G_t^T\right) \Gamma_{t|t} F_t \tag{8.27e}$$

for $t = N, N-1, \ldots, 1$. The Fisher estimates are obtained by initializing the filter with

$$\Gamma_{N|N+1} = 0 \qquad \text{and} \qquad \hat{a}_{N|N+1} = 0. \tag{8.27f}$$

The backwards information filter sequence is computed under the association hypothesis $\Psi_N^{(p-1)}$. Thus, Y_t in (8.27a) is the stacked vector of non-clutter measurements under the hypothesis $\psi_t^{(p-1)}$, and $\mathbb{H}_t$ and $\mathbb{R}_t$ are the corresponding measurement equation matrix and measurement error covariance, respectively.

Collecting the results above yields that the full conditional probability (8.19) is given by

$$p(\psi_t^{(p)} \mid \mathbb{Y}_N, \Psi_{\neg t}^{(p)}) \propto \mathrm{N}(Y_t; \hat{Y}_{t|t-1}, S_t) \frac{p_c(\mu_t - m_t) P_D^{m_t} (1 - P_D)^{n-m_t} V_S^{m_t}}{r_t(m_t)} \times$$
$$|I + P_{t|t}\Gamma_{t|t+1}|^{-1/2} \exp\Big(-\tfrac{1}{2}\|\hat{x}_{t|t}\|^2_{\Gamma_{t|t+1}} + \hat{x}_{t|t}^T \hat{a}_{t|t+1} +$$
$$\tfrac{1}{2}\|\hat{a}_{t|t+1} - \Gamma_{t|t+1}\hat{x}_{t|t}\|^2_{(\Gamma_{t|t+1} + P_{t|t}^{-1})^{-1}}\Big), \quad (8.28)$$

where $\hat{Y}_{t|t-1}$, $\hat{x}_{t|t}$, and $P_{t|t}$ are based on $\psi_t^{(p)}$. It may require to use the matrix inversion lemma on the last term in the exponent of (8.28), and set

$$(\Gamma_{t|t+1} + P_{t|t}^{-1})^{-1} = P_{t|t} - P_{t|t}\Gamma_{t|t+1}(P_{t|t}\Gamma_{t|t+1} + I)^{-1} P_{t|t}$$

whenever $P_{t|t}$ is singular. One iteration of the Gibbs sampler based on (8.28) can thus be computed using an initial backwards information filtering sweep, followed by a forwards sampling iteration that determines the association sequence and sequentially runs a Kalman filter based on this sequence. We label this algorithm the Monte Carlo Data Association (MCDA) algorithm.

Algorithm 8.2 (Monte Carlo Data Association (MCDA))

1. Initialize by picking the initial association $\Psi_N^{(0)}$ randomly or deterministically. Set $p = 1$.
2. Run the backwards information filter (8.27) using $\Psi_N^{(p-1)}$. Store $\Gamma_{t|t+1}$ and $\hat{a}_{t|t+1}$.
3. Initialize the Kalman filter quantities $\hat{x}_{0|-1}$ and $P_{0|-1}$, set $t = 0$, and compute the following forward recursion:
   (a) Run Kalman filter measurement updates under each possible association hypothesis for Y_t, i.e., for $k = 1, \ldots, \pi_t$. Store every $\hat{x}_{t|t}^k$ and $P_{t|t}^k$.
   (b) Compute (8.28) under all π_t hypotheses, normalize this discrete density, and generate a sample from this discrete distribution,
   $$\psi_t^{(p)} \sim p(\psi_t^{(p)} \mid \mathbb{Y}_N, \Psi_{\neg t}^{(p)}).$$
   (c) Set $t := t+1$, run a time update step with the Kalman filter corresponding to the chosen association, store the resulting $\hat{x}_{t+1|t}$ and $P_{t+1|t}$ and use them in the next iteration at item 3(a). If $t > N$, go to item 4 instead.
4. Compute $\mathbb{X}_N^{(p)} = \mathsf{E}\Big(\mathbb{X}_N \mid \mathbb{Y}_N, \Psi_N^{(p)}\Big)$ by fixed interval Kalman smoothing.
5. Output the association sequence $\Psi_N^{(p)}$ and the state sequence $\mathbb{X}_N^{(p)}$. Set $p := p + 1$ and return to item 2.

The algorithm above is iterated M times, and minimum mean square estimates of the state and association sequences are formed according to

$$\widehat{\mathbb{X}}_N = \frac{1}{M}\sum_{i=1}^{M} \mathbb{X}_N^{(i)} \qquad\qquad \widehat{\Psi}_N = \frac{1}{M}\sum_{i=1}^{M} \Psi_N^{(i)},$$

possibly after discarding an initial *burn-in* phase of the algorithm. The association estimate above should be interpreted as a weighted average over some numbering of the hypotheses, possibly followed by a conversion to integer estimates afterwards.

The backwards filtering computation in item 2 of Algorithm 8.2 can actually be retained from the calculations in item 4 by utilizing a specific type of smoothing formula [96]. Mayne [115] formulated the linear smoothing problem as an optimization problem, decomposed into one minimization with respect to past data, and one with respect to future data. Fraser and Potter [69] later suggested that this approach could be interpreted as the combination of two optimal filters: one backwards information filter (8.27), and one regular forward Kalman filter. Let $\hat{x}_{t|t-1}$ and $P_{t|t-1}$ be one step ahead predictions from a standard Kalman filter applied under the same association hypothesis as used in (8.27). The smoothed estimate $\hat{x}_{t|N}$ is computed by combining the quantities $\hat{x}_{t|t-1}$ and $P_{t|t-1}$ with the results of (8.27),

$$\hat{x}_{t|N} = \left(I - P_t\Gamma_{t|t}(P_t\Gamma_{t|t} + I)^{-1}\right)(\hat{x}_{t|t-1} - P_t\hat{a}_{t|t}),$$

see [69, 96]. The estimates produced by a backwards information filter at item 4 can thus be reused at item 2 in the next iteration of the MCDA algorithm.

Algorithm 8.2 can straightforwardly be utilized for MAP, or marginal MAP estimation as well. A simulated annealing algorithm is obtained by sampling from

$$p(\psi_t^{(p)} \mid \mathbb{Y}_N, \Psi_{\neg t}^{(p)})^{\frac{1}{T_p}}$$

where $\{T_p\}$ is a cooling sequence in the sense described in Chapter 6. Alternatively the maximization can be performed deterministically by picking the association that maximizes the full conditional at each iteration, see [61] for similar ideas.

8.2.5 Simulation Result

The simulations follow the setup described in [123]. The target state consists of range, r_t, azimuth, θ_t, and their respective time derivatives, $x_t = (r_t, \dot{r}_t, \theta_t, \dot{\theta}_t)^T$. The true target state trajectory obeys

$$r_t = 50 + 5tT \qquad\qquad \dot{r}_t = 5$$
$$\theta_t = 0.3tT \qquad\qquad \dot{\theta}_t = 0.3$$

with $t = 0, 1, \ldots, 15$ and sampling time $T = 2$. Three measurement models were used,

$$H_i = \begin{bmatrix} 1 + \frac{i-1}{10} & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \qquad \text{for } i \in \{1, 2, 3\},$$

all having identical noise covariance

$$R = \begin{bmatrix} 10^2 & 0 & 0 \\ 0 & 0.1^2 & 0 \\ 0 & 0 & 5^2 \end{bmatrix}$$

The state noise covariance and initial state covariance used in the filters was set to

$$Q = \begin{bmatrix} 0.1 & 0 & 0 & 0 \\ 0 & 0.1 & 0 & 0 \\ 0 & 0 & 0.0125 & 0 \\ 0 & 0 & 0 & 0.0125 \end{bmatrix} \quad \text{and} \quad P_0 = \begin{bmatrix} 200 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The observation volume V_S was defined by

$$\begin{aligned} 25 &\leq r_t \leq 225 \\ 4.5 &\leq \dot{r}_t \leq 5.5 \\ -2 &\leq \theta_t \leq 10. \end{aligned} \tag{8.29}$$

Figure 8.4 shows a simulation case with high probability of correct measurement detection, $P_D = 0.95$, and low clutter intensity, $\lambda V_S = 0.4$. In the figure, solid lines

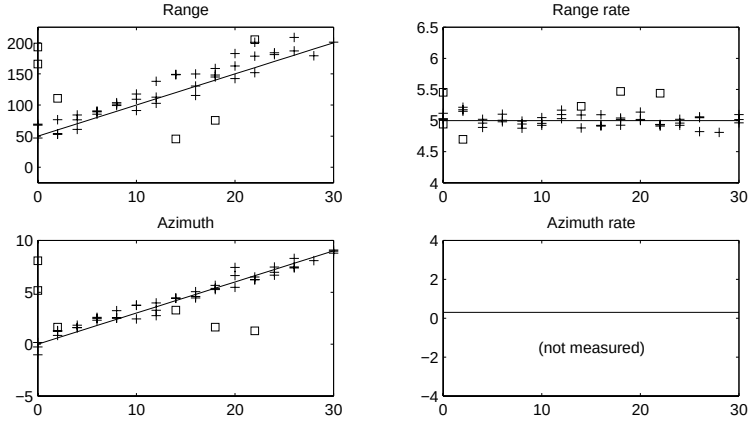


Figure 8.4: Simulation case, $P_D = 0.95$, $\lambda V_S = 0.4$.

show the true target states, pluses indicate measurements origin from the target, and squares indicate clutter points. The typical performance of the algorithms are shown in Figures 8.5–8.7. In all figures, the asterisk $(*)$ indicate the true target state value, the solid line is the estimate, and the dotted line is the $\pm 3\sigma$ confidence region of the estimate. The first measurement was used for initialization of the algorithms. The transients in the MSMF estimates in Figure 8.5 stem from two clutter points in the measurement used for initialization. The iterative passes in the EMDA filter

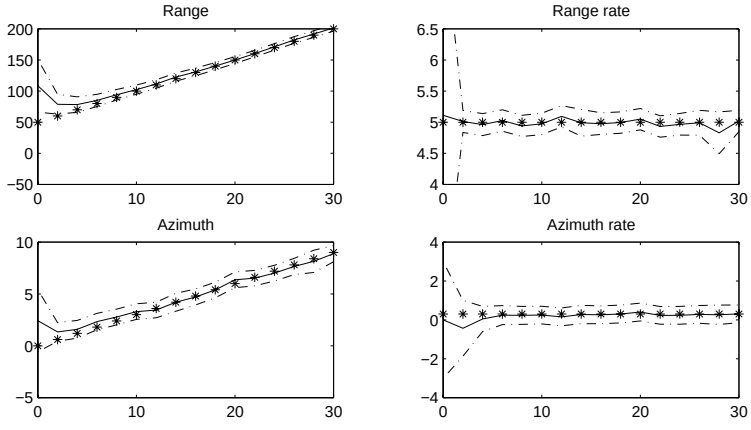


Figure 8.5: Performance of MSMF, $P_D = 0.95$, $\lambda V_S = 0.4$.

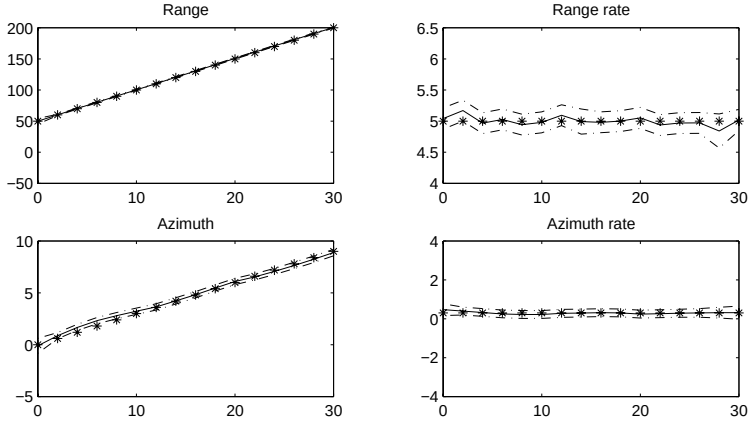


Figure 8.6: Performance of EMDA, $P_D = 0.95$, $\lambda V_S = 0.4$.

manage to compute an almost perfect association, with resulting state estimates depicted in Figure 8.6. The EMDA algorithm converges in less than five iterations in this simulation study, but has considerably higher computational complexity than the MSMF. The results using the MCDA algorithm are shown in Figure 8.7. Since the state estimates of this algorithm are computed by Monte Carlo estimation, no state covariance is available and therefore there are no confidence regions shown in Figure 8.7. The estimates depicted in the figure are the resulting Monte Carlo estimates obtained when running $M = 5$ iterations of the MCDA algorithm, and regarding only the initial iteration as a *burn-in* sample. The computations required to perform one iteration of the MCDA algorithm exceed the computations of the

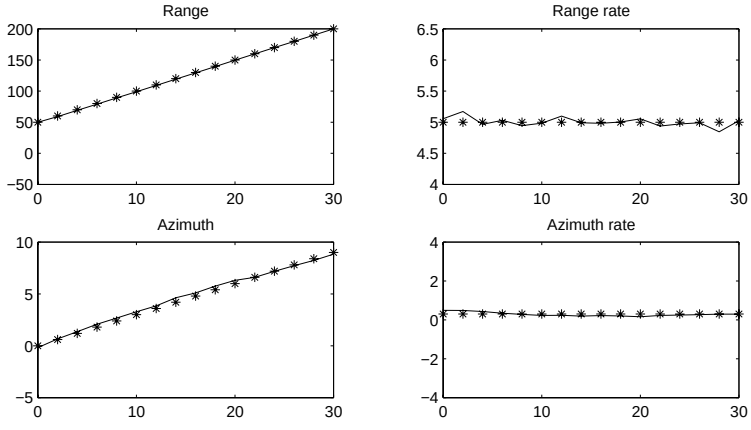


Figure 8.7: Performance of MCDA, $P_D = 0.95$, $\lambda V_S = 0.4$.

EMDA and the MSMF.

A second simulation case with less frequent real measurements and more clutter is depicted in Figure 8.8. In these simulations, on the average two clutter

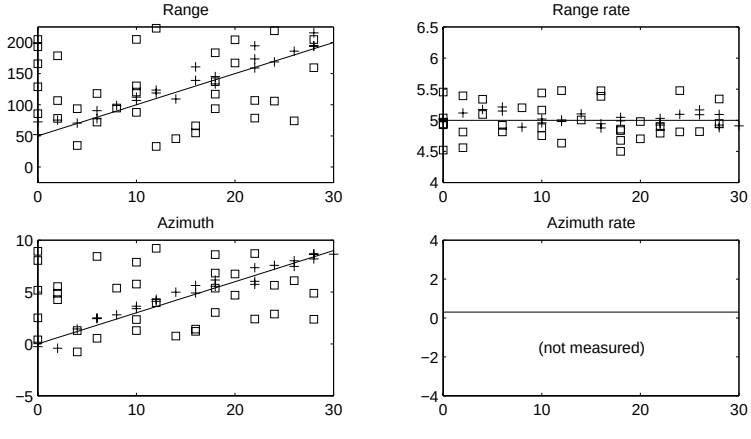


Figure 8.8: Simulation case, $P_D = 0.5$, $\lambda V_S = 2$.

measurements are received in each measurement set, and only half of the real measurement models generate observations. The simulation results obtained with the data in Figure 8.8 are shown in Figure 8.9–8.11. The MSMF filter has greater problems with this dense clutter case, as shown in Figure 8.9. The EMDA algorithm in Figure 8.10 still yields very accurate range estimates while the range rate is less precise. The MCDA performance in Figure 8.11 is almost identical to the

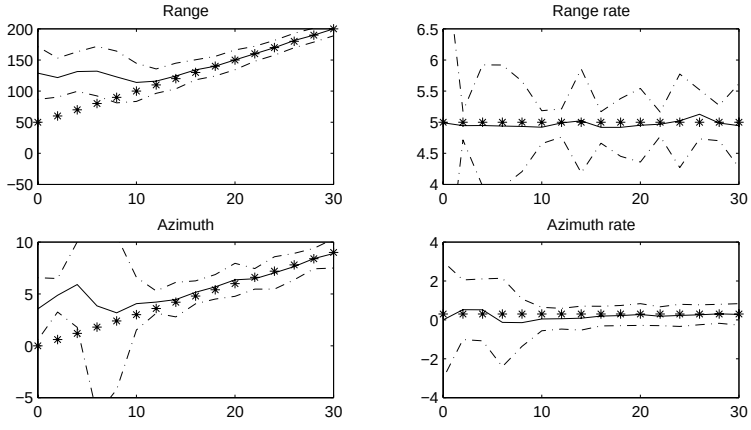


Figure 8.9: Performance of MSMF, $P_D = 0.5$, $\lambda V_S = 2$.

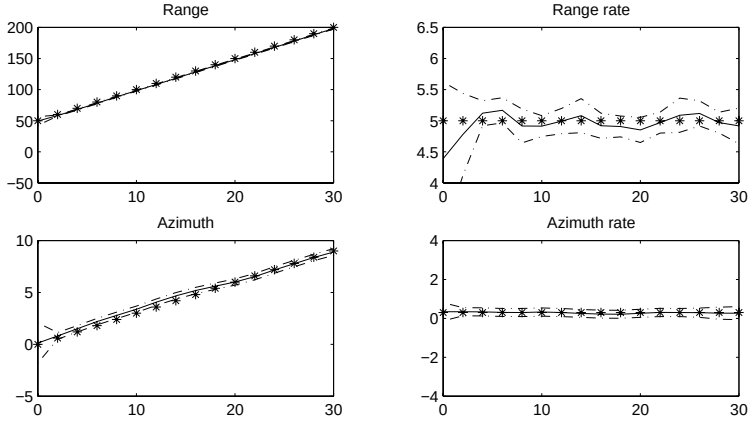


Figure 8.10: Performance of EMDA, $P_D = 0.5$, $\lambda V_S = 2$.

one obtained for the case with less clutter points and more target measurements. The results in Figure 8.11 are computed from a ten iteration run of the MCDA algorithm, discarding the chain output for the initial four iterations.

8.3 Conclusion

In the multiple measurement data association problem, the number of possible association hypotheses may reach very high levels. We have studied three statistical algorithms for this joint measurement association and state estimation prob-

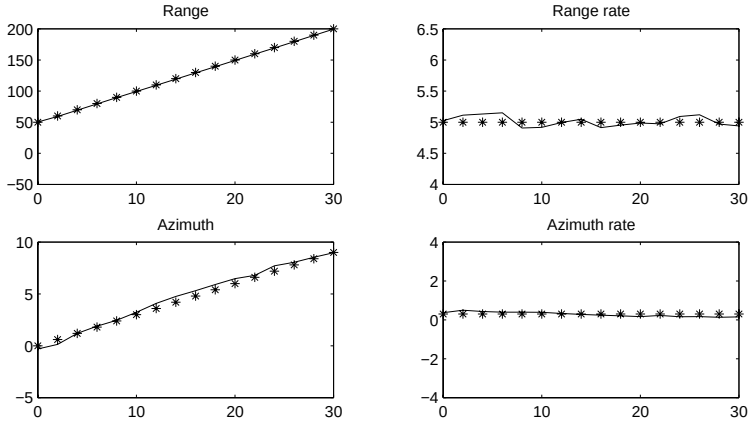


Figure 8.11: Performance of MCDA, $P_D = 0.5$, $\lambda V_S = 2$.

lem. The recursive Multiple Simultaneous Measurement Filter (MSMF) merges the hypotheses at each time iteration, while the Expectation Maximization Data Association (EMDA) applies the EM algorithm for MAP estimation of the association sequence, regarding a batch of the state sequence as unobserved data. A novel Gibbs sampling algorithm for batch processing has been developed for this problem. The Monte Carlo Data Association (MCDA) algorithm has a backward filtering, forward sampling structure and can be utilized to yield either MMSE or MAP estimates of the state and association sequences.

The batch algorithms EMDA and MCDA show superior performance over the recursive MSMF procedure. The primitive initialization used in the simulations degrades the performance of the MSMF considerably when there is clutter in the measurement set. With a more elaborate initialization this effect could perhaps be attenuated. The MSMF would also gain in accuracy with the introduction of measurement gating [7]. Neither the EMDA nor the MCDA seems to suffer from the initialization problems, though. These algorithms yield reliable state estimates, even in the simulation case of very dense clutter.

Appendix

8.A Expectation Maximization for Segmentation

This appendix contains a derivation of the EM algorithm for segmentation presented in Section 8.1.1.

We introduce the stacked stacked vectors of N random variables from (8.2)

$$\mathbb{X} = [x_1^T, x_2^T, \dots, x_N^T]^T \qquad \mathbb{U} = [x_1^T, w_{1,\delta}^T, \dots, w_{N-1,\delta}^T]^T$$
$$\mathbb{E} = [e_{1,\delta}^T, e_{2,\delta}^T, \dots, e_{N,\delta}^T]^T$$

where the noise vectors $\mathbb{U}$ and $\mathbb{E}$ explicitly depend on the segmentation sequence Δ. Using this notation, the estimation model (8.2) can be written compactly as

$$\begin{aligned} \mathbb{X} &= A_\Delta \mathbb{U} \\ \mathbb{Y} &= B_\Delta \mathbb{X} + \mathbb{E} \end{aligned} \tag{8.A.30}$$

where matrices A_Δ and B_Δ are block matrices formed by the state space matrices of the model (8.2). Direct inspection yields that

$$A_\Delta = \begin{bmatrix} I & 0 & \dots & 0 \\ W_1^1 & I & \dots & 0 \\ W_1^2 & W_2^2 & \dots & 0 \\ \vdots & & \ddots & \vdots \\ & & & 0 \\ W_1^{N-1} & \dots & W_{N-1}^{N-1} & I \end{bmatrix} \begin{bmatrix} I & 0 & \dots & 0 \\ 0 & G_{1,\delta} & \dots & 0 \\ 0 & 0 & G_{2,\delta} & 0 \\ \vdots & & \ddots & \vdots \\ & & & 0 \\ 0 & \dots & 0 & G_{N-1,\delta} \end{bmatrix}$$

where zeros denote zero matrices of appropriate dimension, and the state transition matrix is defined as

$$W_t^l = F_{l,\delta} F_{l-1,\delta} \cdots F_{t,\delta}.$$

Furthermore,

$$B_\Delta = \operatorname{diag}(H_{1,\delta}, H_{2,\delta}, \dots, H_{N,\delta})$$

where $\mathrm{diag}(A_1, A_2, \ldots, A_M)$ denotes a matrix with blocks A_i along the diagonal. The statistical properties of the noises in (8.A.30) are compactly given in the expression

$$\mathsf{E}\left\{\begin{bmatrix}\mathbb{U}\\ \mathbb{E}\end{bmatrix}\begin{bmatrix}\mathbb{U}^T & \mathbb{E}^T & 1\end{bmatrix}\right\} = \begin{bmatrix}Q_\Delta & 0 & \left[\begin{smallmatrix}\hat{x}_1\\ 0\end{smallmatrix}\right]\\ 0 & R_\Delta & 0\end{bmatrix} \tag{8.A.31}$$

where both Q_Δ and R_Δ are block diagonal,

$$\begin{aligned} Q_\Delta &= \mathrm{diag}\,(P_1, Q_{1,\delta}, Q_{2,\delta}, \ldots, Q_{N-1,\delta})\\ R_\Delta &= \mathrm{diag}\,(R_{1,\delta}, R_{2,\delta}, \ldots, R_{N,\delta})\,. \end{aligned}$$

Eliminating the states $\mathbb{X}$ in (8.A.30) yields

$$\mathbb{Y} = B_\Delta A_\Delta \mathbb{U} + \mathbb{E}. \tag{8.A.32}$$

In the EM algorithm of Section 8.1.1, we treat the segmentation sequence Δ as the parameters and the vector $\mathbb{U}$ as the unobserved data in the model (8.A.32). With a Bayesian framework each EM pass p is defined by

$$\Delta_{p+1} = \arg\max_{\Delta} \mathsf{E}\left(\log p(\mathbb{Y}, \mathbb{U}, \Delta) \,|\, \mathbb{Y}, \Delta_p\right) \tag{8.A.33}$$

where the joint density can be divided into three factors

$$p(\mathbb{Y}, \mathbb{U}, \Delta) = p(\mathbb{Y} \,|\, \mathbb{U}, \Delta) p(\mathbb{U} \,|\, \Delta) p(\Delta). \tag{8.A.34}$$

The first and second factors are given by (8.A.32) and (8.A.31) above,

$$\begin{aligned} p(\mathbb{Y} \,|\, \mathbb{U}, \Delta) &= \mathrm{N}\left(\mathbb{Y}; B_\Delta A_\Delta \mathbb{U}, R_\Delta\right)\\ p(\mathbb{U} \,|\, \Delta) &= \mathrm{N}\left(\mathbb{U}; \left[\begin{smallmatrix}\hat{x}_1\\ 0\end{smallmatrix}\right], Q_\Delta\right). \end{aligned}$$

The last factor is the Bayesian prior for the segmentation sequence.

Denoting the quadratic norm $x^T A x$ by $\|x\|_A^2$, and the determinant by $|A|$, the logarithm of (8.A.34) is

$$\begin{aligned} \log p(\mathbb{Y}, \mathbb{U}, \Delta) = \log p(\Delta) - \tfrac{1}{2}\|\mathbb{Y} - B_\Delta A_\Delta \mathbb{U}\|^2_{R_\Delta^{-1}} - \tfrac{1}{2}\log|R_\Delta| - \tfrac{1}{2}\log|Q_\Delta| -\\ - \tfrac{1}{2}\|\mathbb{U} - \left[\begin{smallmatrix}\hat{x}_1\\ 0\end{smallmatrix}\right]\|^2_{Q_\Delta^{-1}} - \tfrac{(N-1)n_u + N n_y + n_x}{2}\log(2\pi). \end{aligned} \tag{8.A.35}$$

Since the maximization in (8.A.33) is performed over the segmentation sequence Δ only terms involving the segmentation parameters need to be retained from (8.A.35) when the conditional expectation is evaluated. Hence, removing terms and positive factors independent of the segmentation sequence we define the return function

$$\begin{aligned} J(\mathbb{Y}, \mathbb{U}, \Delta) = 2\log p(\Delta) - \|\mathbb{Y} - B_\Delta A_\Delta \mathbb{U}\|^2_{R_\Delta^{-1}} - \log|R_\Delta| - \log|Q_\Delta| -\\ - \|\mathbb{U} - \left[\begin{smallmatrix}\hat{x}_1\\ 0\end{smallmatrix}\right]\|^2_{Q_\Delta^{-1}} \end{aligned} \tag{8.A.36}$$

and use the auxiliary function

$$\mathcal{Q}(\Delta, \Delta_p) = \mathsf{E}_{\mathbb{U}} \left(J(\mathbb{Y}, \mathbb{U}, \Delta) \,|\, \mathbb{Y}, \Delta_p \right).$$

The conditional mean and covariance of the vector $\mathbb{U}$, given by

$$\begin{aligned} \hat{\mathbb{U}}_p &= \mathsf{E} \left(\mathbb{U} \,|\, \mathbb{Y}, \Delta_p \right) \\ \mathbb{S}_p &= \mathsf{E} \left((\mathbb{U} - \hat{\mathbb{U}}_p)(\mathbb{U} - \hat{\mathbb{U}}_p)^T \,|\, \mathbb{Y}, \Delta_p \right), \end{aligned} \tag{8.A.37}$$

can be obtained using standard linear estimation theory on the model (8.A.32) after inserting the segmentation sequence Δ_p. Taking the conditional expectation of (8.A.36), inserting (8.A.37) and completing the squares yield

$$\begin{aligned} \mathcal{Q}(\Delta, \Delta_p) = 2 \log p(\Delta) - \| \mathbb{Y} - B_\Delta A_\Delta \hat{\mathbb{U}}_p \|^2_{R_\Delta^{-1}} - \log |R_\Delta| - \log |Q_\Delta| - \\ - \| \hat{\mathbb{U}}_p - \left[\begin{smallmatrix} \hat{x}_1 \\ 0 \end{smallmatrix} \right] \|^2_{Q_\Delta^{-1}} - \operatorname{tr} \left(\left(A_\Delta^T B_\Delta^T R_\Delta^{-1} B_\Delta A_\Delta + Q_\Delta^{-1} \right) \mathbb{S}_p \right) \end{aligned} \tag{8.A.38}$$

where the linearity of the trace operator $\operatorname{tr}(\cdot)$ and the relation $\|x\|^2_A = \operatorname{tr}(Axx^T)$ has been used repeatedly.

Standard fixed interval linear state smoothing can be used to compute the return (8.A.38). Define a notation for the conditional mean and covariance of the complete state sequence

$$\begin{aligned} \hat{\mathbb{X}}_p &= \mathsf{E} \left(\mathbb{X} \,|\, \mathbb{Y}, \Delta_p \right) \\ \mathbb{P}_p &= \mathsf{E} \left((\mathbb{X} - \hat{\mathbb{X}}_p)(\mathbb{X} - \hat{\mathbb{X}}_p)^T \,|\, \mathbb{Y}, \Delta_p \right). \end{aligned} \tag{8.A.39}$$

From (8.A.30) we have that

$$\mathbb{X} = A_\Delta \mathbb{U} \qquad \mathbb{U} = A_\Delta^\dagger \mathbb{X}$$

where $M^\dagger = (M^T M)^{-1} M^T$ is the Moore-Penrose pseudo inverse. Inserting this into the return (8.A.36) yields

$$\begin{aligned} J(\mathbb{Y}, \mathbb{X}, \Delta) = 2 \log p(\Delta) - \| \mathbb{Y} - B_\Delta \mathbb{X} \|^2_{R_\Delta^{-1}} - \log |R_\Delta| - \log |Q_\Delta| - \\ - \| A_\Delta^\dagger \mathbb{X} - \left[\begin{smallmatrix} \hat{x}_1 \\ 0 \end{smallmatrix} \right] \|^2_{Q_\Delta^{-1}} \end{aligned}$$

taking conditional expectation, inserting (8.A.39) and completing the squares we have that

$$\begin{aligned} \mathcal{Q}(\Delta, \Delta_p) = 2 \log p(\Delta) - \| \mathbb{Y} - B_\Delta \hat{\mathbb{X}}_p \|^2_{R_\Delta^{-1}} - \log |R_\Delta| - \log |Q_\Delta| - \\ - \| A_\Delta^\dagger \hat{\mathbb{X}}_p - \left[\begin{smallmatrix} \hat{x}_1 \\ 0 \end{smallmatrix} \right] \|^2_{Q_\Delta^{-1}} - \operatorname{tr} \left(\left(B_\Delta^T R_\Delta^{-1} B_\Delta + A_\Delta^{\dagger T} Q_\Delta^{-1} A_\Delta^\dagger \right) \mathbb{P}_p \right). \end{aligned} \tag{8.A.40}$$

It is straightforward to verify that the pseudo inverse of A_Δ has a strong diagonal structure,

$$A_\Delta^\dagger = \begin{bmatrix} I & 0 & \dots & & 0 \\ -G_{1,\delta}^\dagger F_{1,\delta} & G_{1,\delta}^\dagger & 0 & \dots & 0 \\ 0 & -G_{2,\delta}^\dagger F_{2,\delta} & G_{2,\delta}^\dagger & 0 \dots & 0 \\ \vdots & & & & \vdots \\ & & & & 0 \\ 0 & \dots & 0 & -G_{N-1,\delta}^\dagger F_{N-1,\delta} & G_{N-1,\delta}^\dagger \end{bmatrix}.$$

Furthermore, B_Δ, R_Δ^{-1} and Q_Δ^{-1} are all block diagonal which means that each norm and matrix trace in (8.A.40) can be written as a sum of norms and matrix traces of smaller matrices. Introducing a notation for the smoothed state estimate and (cross) covariance

$$\begin{aligned} \hat{x}_t^p &= \mathsf{E}\left(x_t \,|\, \mathbb{Y}, \Delta_p\right) \\ P_{l,t}^p &= \mathsf{E}\left((x_l - \hat{x}_l^p)(x_t - \hat{x}_t^p)^T \,|\, \mathbb{Y}, \Delta_p\right), \end{aligned}$$

it follows that the return function (8.A.40) can be written

$$\begin{aligned} \mathcal{Q}(\Delta, \Delta_p) = \sum_{t=1}^{N} \Big(2\delta_t \log\left(\tfrac{1-q}{q}\right) - \|y_t - H_{t,\delta}\hat{x}_t^p\|^2_{R_{t,\delta}^{-1}} - \operatorname{tr}\left(H_{t,\delta}^T R_{t,\delta}^{-1} H_{t,\delta} P_{t,t}^p\right) - \\ - \log|R_{t,\delta}|\Big) - \sum_{t=1}^{N-1} \Big(\|G_{t,\delta}^\dagger(\hat{x}_{t+1}^p - F_{t,\delta}\hat{x}_t^p)\|^2_{Q_{t,\delta}^{-1}} + \log|Q_{t,\delta}| + \\ \operatorname{tr}\left(G_{t,\delta}^{\dagger T} Q_{t,\delta}^{-1} G_{t,\delta}^\dagger (P_{t+1,t+1}^p - 2F_{t,\delta}P_{t,t+1}^p + F_{t,\delta}P_{t,t}^p F_{t,\delta}^T)\right)\Big), \end{aligned} \quad (8.A.41)$$

where the terms independent of Δ have been removed from the expression, and the Bernoulli prior (8.4),

$$\log p(\Delta) = \sum_{t=1}^{N} \delta_t \log\left(\tfrac{1-q}{q}\right),$$

has been inserted.

9

Conclusive Remarks

The Bayesian paradigm for statistical inference yields a natural framework for recursive nonlinear estimation problems. All information gained about the states from initial to present time is in this framework condensed by the posterior filter density. The conceptual recursive update of the posterior density is simple in structure, yet impossible to implement when then problem is of nonlinear and non-Gaussian character.

The pure notion of the existence of the optimal solution is valuable in the hunt for high performing algorithms. Rather than blindly turn to model approximations, we propose to utilize the conceptual update and approximately implement it as efficiently as possible, e.g., using a point-mass approach. The main difference between model and solution approximations is that in the latter scheme, it is possible to obtain a graceful degradation of the optimal solution. A tradeoff between estimation accuracy and implementational complexity is found by controlling the grid mesh resolution in a numerical implementation.

The Bayesian approach to recursive estimation has proven to be utterly successful for the terrain navigation application. The results from the work on this application shows that complex problems of this type can be handled with great success using approximative Bayesian inference. The point-mass implementation has shown to yield high performance in the terrain navigation application, both on simulated but realistic measurements and on actual field test data. One of the future challenges for this application is to maximize the utilization of the point-mass

description for the integration with the inertial navigation system.

The Cramér-Rao bounds to recursive estimation provides an appealing approach to evaluation of suboptimal algorithms. The recursive expressions for Cramér-Rao bounds to nonlinear filtering has provided us with arguments that the suboptimal point-mass implementation of the terrain navigation application has near optimal performance. The way we estimate both the bounds and the algorithm performance in Monte Carlo simulations have shown to yield a concrete tool for verification of suboptimal algorithms. Both the parametric and the Bayesian, i.e., posterior, bounds can be utilized in such Monte Carlo evaluations.

Practical implementation of recursive Bayesian estimation problems has shown to be tightly connected to function approximation and numerical integration. A general combination of these concepts was pursued in the ideas outlined about a wavelet multiresolution approach to spatial grid adaption. Even though a great deal of time has been spent on investigations into this approach, the resulting conclusion from this work is rather discouraging and the concepts are only given minor space in this presentation. Guidelines about adaptive grid methods for general nonlinear recursive estimation will inevitably be very hard to set. We believe that each recursive estimation problem should instead be investigated and solved with the methods best suited for that application. In some cases, an adaptive mesh will prove useful, in others a uniform mesh will suffice. The wavelet multiresolution techniques provides a uniform approach to function approximation and numerical integration. However, the implementational difficulties are currently too overwhelming for pursuing this approach in a practical application.

The Monte Carlo filters rely on a straightforward simulation strategy to determine a spatially adaptive grid, well suited for the problem at hand. The particle filters are simple in structure and demand no design of a grid mesh propagation or resampling scheme. In the comparative simulations of the terrain navigation application we have seen that the particle filters and the point-mass approach reach the same level of performance. Future work will involve applying the particle filters to terrain navigation when the absolute altitude is biased, or directly to the integration of inertial navigation and terrain navigation.

Simulation based methods for recursive estimation is currently a very active research field which interests both pure statisticians and people form the field of signal processing. Different approaches to enhance the performance of the basic particle filters are constantly appearing in the literature. One main advantage with the simulation based algorithms is their ease of implementation. They also extend to higher dimensional problems with only minor implementational effort. We believe that these issues will make the basic Monte Carlo filters reviewed in this work the future workhorses of nonlinear recursive estimation.

Bibliography

[1] D.L. Alspach and H.W. Sorenson. Nonlinear Bayesian estimation using Gaussian sum approximation. *IEEE Transactions on Automatic Control*, 17:439–448, 1972.

[2] B.D.O. Anderson and J.B. Moore. *Optimal Filtering*. Prentice Hall, Englewood Cliffs, NJ, 1979.

[3] C. Andrieu, A. Doucet, and P. Duvaut. Joint Bayesian detection and estimation of sinusoids embedded in noise. In *Proc. IEEE Conf. Acoustics, Speech and Signal Processing (ICASSP)*, 1998.

[4] C. Andrieu, A. Doucet, S.J. Godsill, and W.J. Fitzgerald. An introduction to the theory and applications of simulation based computational methods in Bayesian signal processing. Technical Report CUED/F-INFENG/TR.324, Department of Engineering, University of Cambridge, UK, 1998.

[5] D. Avitzour. Stochastic simulation Bayesian approach to multitarget tracking. *IEE Proc. on Radar, Sonar and Navigation*, 142(2), 1995.

[6] W. Baker and R. Clem. Terrain contour matching [TERCOM] primer. Technical Report ASP-TR-77-61, Aeronautical Systems Division, Wright-Patterson AFB, Aug. 1977.

[7] Y. Bar-Shalom and T. Fortmann. *Tracking and Data Association*, volume 179 of *Mathematics in Science and Engineering*. Academic Press, 1988.

[8] L.E. Baum, T. Petrie, G. Soules, and N. Weiss. A maximization technique occuring in the statistical analysis of probabilistic functions of Markov chains. *The Annals of Mathematical Statistics*, 41:164–171, 1970.

[9] T.R. Bayes. An essay towards solving a problem in the doctrine of chances. *Phil. Trans. Roy. Soc. London*, 53(370), 1763. Reprinted in *Biometrica*, 45(293), 1958.

[10] E.R. Beadle and P.M. Djurić. A fast weighted Bayesian bootstrap filter for nonlinear model state estimation. *IEEE Transactions on Aerospace and Electronic Systems*, 33(1):338–343, 1997.

[11] E.R. Beadle and P.M. Djurić. Parameter estimation for non-Gaussian autoregressive processes. In *Proc. IEEE Conf. Acoustics, Speech and Signal Processing (ICASSP)*, 1997.

[12] E.R. Beadle and P.M. Djurić. Uniform random parameter generation of stable minimum-phase real ARMA (p, q) processes. *IEEE Signal Processing Letters*, 4(9), 1997.

[13] V.E. Beneš. Exact finite-dimensional filters for certain diffusions with nonlinear drift. *Stochastics*, 5:65–92, 1981.

[14] J.O. Berger. *Statistical Decision Theory and Bayesian Analysis*. Springer-Verlag, 2nd edition, 1985.

[15] N. Bergman. A Bayesian approach to terrain-aided navigation. In *Proc. of SYSID'97, 11th IFAC Symposium on System Identification*, pages 1531–1536. IFAC, 1997.

[16] N. Bergman. *Bayesian Inference in Terrain Navigation*. Linköping Studies in Science and Technology. Thesis No 649, 1997.

[17] N. Bergman. On the Cramer-Rao bound for terrain-aided navigation. Technical Report LiTH-ISY-R-1970, Department of Electrical Engineering, Linköpings University, 1997.

[18] N. Bergman. Deterministic and stochastic Bayesian methods in terrain navigation. In *Proc. 37:th IEEE Conf. on Decision and Control*, 1998.

[19] N. Bergman. Expectation maximization segmentation. Technical Report LiTH-ISY-R-2067, Department of Electrical Engineering, Linköping University, 1998.

[20] N. Bergman. An interpolating wavelet filter for terrain navigation. In *Proc. conf. on multisource-multisensor information fusion*, pages 251–258, 1998.

[21] N. Bergman. Terrain navigation using sequential Monte Carlo methods. In A. Doucet, J.F.G. de Freitas, and N.J. Gordon, editors, *Sequential Monte Carlo methods in practice*. Cambridge University Press, 1999. To appear.

[22] N. Bergman and F. Gustafsson. Three statistical batch algorithms for tracking manoeuvring targets. In *Proc. 5th European Control Conference*, Karlsruhe, Germany, 1999.

[23] N. Bergman, L. Ljung, and F. Gustafsson. Point-mass filter and Cramér-Rao bound for terrain-aided navigation. In *Proc. 36:th IEEE Conf. on decision and control*, pages 565–570, 1997.

[24] N. Bergman, L. Ljung, and F. Gustafsson. Terrain navigation using Bayesian statistics. *IEEE Control Systems Magazine*, 19(3), June 1999.

[25] N. Bergman and P. Tichavský. Two Cramér-Rao bounds for terrain-aided navigation. *IEEE Transactions on Aerospace and Electronic Systems*, 1999. In review.

[26] G. Beylkin. On the fast algorithm for multiplication of functions in the wavelet bases. Technical report, Prog. in Appl. Math., Univ. of Colorado at Boulder, Boulder, CO 80309-0526, jun 1992.

[27] G. Beylkin. Wavelets and fast numerical algorithms. In I. Daubechies, editor, *Different Perspectives on Wavelets*, volume 47 of *Proceedings of Symposia in Applied Mathematics*, pages 89–117. American Math. Soc., Providence, RI, 1993. From an American Math. Soc. short course, Jan. 11–12, 1993, San Antonio, TX.

[28] G. Beylkin, R. Coifman, and V. Rokhlin. Fast wavelet transforms and numerical algorithms. *Comm. in Pure and Applied Math.*, 44:141–183, 1991.

[29] S.S. Blackman. *Multiple-target tracking with radar applications*. Artech House, Norwood, MA, 1986.

[30] E. Bølviken, P.J. Acklam, N. Christophersen, and J-M Størdal. Monte Carlo filters for non-linear state estimation. Technical report, University of Oslo, 1997.

[31] D. Boozer and J. Fellerhoff. Terrain-Aided Navigation Test Results in the AFTI/F-16 Aircraft. *Journal of The Institute of Navigation*, 35(2):161–175, Summer 1988.

[32] B.Z. Borobsky and M. Zakai. A lower bound on the estimation error for Markov processes. *IEEE Transactions on Automatic Control*, 20(6):785–788, 1975.

[33] G. Box and G. Tiao. *Bayesian Inference in Statistical Analysis*. John Wiley, 1992.

[34] R.S. Bucy and K.D. Senne. Digital synthesis of nonlinear filters. *Automatica*, 7:287–298, 1971.

[35] D.M. Buende and P. Girardi. A target identification comparison of Bayesian and Dempster-Shafer multisensor fusion. *IEEE Tr. Man and Cybernetics*, 27 (5):569–577, 1997.

[36] Z. Cai, F. Le Gland, and H. Zhang. An adaptive local refinement method for nonlinear filtering. Technical Report No 2679, Unité de rechereche INRIA Rennes, 1995.

[37] J. Carpenter, P. Clifford, and P. Fearnhead. An improved particle filter for non-linear problems. In *IEE Proceedings on Radar and Signal Processing*, 1998. In press.

[38] J. Carpenter, P. Clifford, and P. Fearnhead. Building robust sumilation-based filters for evolving data sets. Technical report, Department of Statistics, University of Oxford, 1999.

[39] H. Carvalho, P. Del Moral, A. Monin, and G. Salut. Optimal nonlinear filtering in GPS/INS integration. *IEEE Transactions on Aerospace and Electronic Systems*, 33(3), 1997.

[40] K.L. Chung. *A course in probability theory*. Academic Press, 1968.

[41] D. Crisan, P. Del Moral, and T. Lyons. Discrete filtering using branching and interacting particle systems. Technical Report 1-98, Laboratoire de Statistique et Probabilités, Université Paul Sabatier, Toulouse, 1998.

[42] I. Daubechies. *Ten Lectures on Wavelets*. SIAM, 1992.

[43] F.E. Daum. New exact nonlinear filters. In J.C. Spall, editor, *Bayesian Analysis of Time Series and Dynamic Models*, volume 94 of *Statistics, textbooks and monographs*, chapter 8, pages 199–226. Marcel Dekker inc., New York, 1988.

[44] P. Davies. The F-16 digital terrain system. In *IEE 1995 Colloquium on Terrain databases and their use in navigation and collision avoidance*. IEE, 1995.

[45] P.J. Davis and P. Rabinowitz. *Numerical Integration*. Blaisdell Publishing Company, 1967.

[46] P.J. Davis and P. Rabinowitz. *Methods of Numerical Integration*. Academic Press, 2nd edition, 1984.

[47] R.J.P. de Figueiredo and Y.G. Jan. Spline filters. In *Proc. 2nd symp. on nonlinear estimation theory and its applications*, pages 127–138, 1971.

[48] J.F.G. de Freitas, M. Niranjan, A.H. Gee, and A. Doucet. Sequential Monte Carlo methods for optimization of neural network models. Technical Report CUED/F-INFENG/TR.324, Department of Engineering, University of Cambridge, UK, 1998.

[49] P. Del Moral. Measure-valued processes and interacting particle systems. Application to nonlinear filtering problems. *The Annals of Applied Probability*, 8(2):438–495, 1998.

[50] A.P. Dempster. New methods for reasoning towards posterior distributions based on sample data. *Ann. Math. Stat.*, 34:355–374, 1966.

[51] A.P. Dempster. Upper and lower probabilites induced by a multivalued mapping. *Ann. Math. Stat.*, 38:325–339, 1967.

[52] A.P. Dempster. A generalization of Bayesian inference. *Journal of the Royal Statistical Society*, 30:202–247, 1968.

[53] A.P Dempster, N.M. Laird, and D.B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, pages 1–38, 1977.

[54] G. Deslauriers and S. Dubuc. Symmetric iterative interpolation processes. *Constructive Approximation*, 5(1):49–68, 1989.

[55] P.M. Djurić, S.J. Godsill, W.J. Fitzgerald, and P.J.W. Rayner. Detection and estimation of signals by reversible jump Markov chain Monte Carlo computations. In *Proc. IEEE Conf. Acoustics, Speech and Signal Processing (ICASSP)*, 1998.

[56] P.C. Doerschuk. Cramer-Rao bounds for discrete-time nonlinear filtering problems. *IEEE Transactions on Automatic Control*, 40(8):1465–1469, 1995.

[57] D.L. Donoho. De-noising by soft-thresholding. Technical Report 409, Dept. of Statistics, Stanford University, 1992.

[58] D.L. Donoho. Interpolating wavelet transforms. Technical Report 408, Dept. of Statistics, Stanford University, 1992.

[59] D.L. Donoho. Smooth wavelet decompositions with blocky coefficient kernels. In L.L. Shumaker and G. Webb, editors, *Recent Advances in Wavelet Analysis*, pages 259–308. Academic Press, New York, 1993.

[60] A. Doucet. On sequential simulation-based methods for Bayesian filtering. Technical Report CUED/F-INFENG/TR.310, Signal Processing Group, Department of Enigeering, University of Cambridge, 1998.

[61] A. Doucet and C. Andrieu. Iterative algorithms for state estimation of jump Markov linear systems. *IEEE Transactions on Signal Processing*, 1999. In review.

[62] A. Doucet, A. Logothetis, and V. Krishnamurthy. Stochastic sampling algorithms for state space estimation of jump Markov linear systems. *IEEE Transactions on Automatic Control*, 1999. Accepted for publ.

[63] R.J. Elliott, L. Aggoun, and J.B. Moore. *Hidden Markov Models – Estimation and Control*. Springer Verlag, 1997.

[64] R. Enns and D. Morrell. Terrain-aided navigation using the Viterbi algorithm. *Journal of Guidance, Control and Dynamics*, 18(6):1444–1449, November-December 1995.

[65] J. Evans and V. Krishnamurthy. Optimal filtering of doubly stochastic autoregressive processes. *Automatica*, 35(2):241–250, 1999.

[66] P. Fearnhead. *Sequential Monte Carlo methods in filter theory*. PhD thesis, University of Oxford, 1998.

[67] N. Ferm. Identity fusion and classification of aircraft in a multisensor environment. Masters Thesis LiTH-ISY-EX-1985, Department of Electrical Engineering, Linköping University, 1998.

[68] D.A.S. Fraser. *The Structure of Inference*. John Wiley, New York, 1968.

[69] D.C. Fraser and J.E. Potter. The optimum linear smoother as a combination of two optimum linear filters. *IEEE Transactions on Automatic Control*, 14 (4):387–390, 1969.

[70] J.I. Galdos. A Cramér-Rao bound for discrete-time nonlinear filtering problems. *IEEE Transactions on Automatic Control*, 25(1):117–119, 1980.

[71] S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6:731–741, 1984.

[72] J. Geweke. Bayesian inference in econometrics models using Monte Carlo integration. *Econometrica*, 57(6):1317–1339, 1989.

[73] W.R. Gilks, S. Richardson, and D.J. Spiegelhalter, editors. *Markov Chain Monte Carlo in Practice*. Chapman & Hall, 1996.

[74] S. Goedecker. *Wavelets and their application for the solution of partial differential equations in physics*. Presses Polytechniques et Universitaires Romandes, 1998.

[75] J.P. Golden. Terrain contour matching (TERCOM): a cruise missile guidance aid. In T. F. Wiener, editor, *Image Processing for Missile Guidance*, volume 238, pages 10–18. The Society of Photo-Optical Instrumentation Engineers (SPIE), July 1980.

[76] N. Gordon. A hybrid Bootstrap filter for target tracking in clutter. *IEEE Transactions on Aerospace and Electronic Systems*, 33(1):353–358, 1997.

[77] N. Gordon and A. Whitby. A Bayesian approach to target racking in the presence of glint. In *Proc. of SPIE, Signal and Data Processing of Small Targets*, volume 2561, pages 472–483, 1995.

[78] N.J. Gordon, D.J. Salmond, and A.F.M. Smith. A novel approach to nonlinear/non-Gaussian Bayesian state estimation. In *IEE Proceedings on Radar and Signal Processing*, volume 140, pages 107–113, 1993.

[79] A. Graham. *Kronecker Products and Matrix Calculus With Applications*. Ellis Horwood Limited, Chichester, England, 1981.

[80] F. Gustafsson. The marginalized likelihood ratio test for detecting abrupt changes. *IEEE Transactions on Automatic Control*, 41(1):66–78, 1996.

[81] Haario and E. Sackman. Simulated annealing in general state space. *Adv. Appl. Prob.*, 23:866–893, 1991.

[82] D.L. Hall. *Mathematical techniques in multi-sensor data fusion*. Artech House, Boston, 1992.

[83] J.E. Handschin. Monte Carlo techniques for prediction and filtering of non-linear stochastic processes. *Automatica*, 6:555–563, 1970.

[84] J.E. Handschin and D.Q. Mayne. Monte Carlo techniques to estimation the conditional expectation in multi-stage non-linear filtering. *Int. J. Cont.*, 9 (5):547–559, 1969.

[85] W.K. Hastings. Monte Carlo simulation methods using Markov chains and their applications. *Biometrica*, 57:97–109, 1970.

[86] A.J. Henley. Terrain aided naviagtion – current status, techniques for flat terrain and reference data requirements. In *Proc. of IEEE 1990 Position, Location and Navigation Symposium (PLANS)*, Las Vegas, Mar. 1990. IEEE.

[87] C. Hewitt. The use of terrain databases for avionic systems. In *IEE 1995 Colloquium on Terrain databases and their use in navigation and collision avoidance*. IEE, 1995.

[88] J.A. Hollowell. Heli/SITAN: A terrain referenced navigation algorithm for helicopters. Las Vegas, Mar. 1990. IEEE Pos., Loc. and Nav. Symp. (PLANS).

[89] M. Holmström. *Wavelet Based Methods for Time Dependent PDEs*. PhD thesis, Dept. of Scientific Computing, Uppsala University, 1997.

[90] R.A. Horn and C.R. Johnson. *Topics in Matrix Analysis*. Cambridge University Press, 1991.

[91] L.D. Hostetler. Optimal terrain-aided navigation systems. In *AIAA Guidance and Control Conference*, Palo Alto, CA, Aug. 1978.

[92] C. Hwang. Laplace's method revisited: weak convergence of probability measures. *Ann. Prob.*, 8:1177–1182, 1980.

[93] A. Isaksson, F. Gustafsson, and N. Bergman. Pruning versus merging in Kalman filter banks for manoeuvre tracking. *IEEE Transactions on Aerospace and Electronic Systems*, 1999. Accepted for publication.

[94] A. Jazwinski. *Stochastic Process and Filtering Theory*, volume 64 of *Mathematics in Science and Engineering*. Academic Press, New York, 1970.

[95] T. Kailath. *Linear systems*. Prentice-Hall, Englewood Cliffs, N.J., 1980.

[96] T. Kailath, A. Sayed, and B. Hassibi. *State Space Estimation Theory*. To appear, 1999.

[97] G. Kaiser. *A Friendly Guide to Wavelets*. Birkhäuser Verlag, 1994.

[98] R. E. Kalman. A new approach to linear filtering and prediction problems. *Trans. AMSE, J. Basic Engineering*, 82:35–45, 1960.

[99] M. Kayton and W. Fried, editors. *Avionics Navigation Systems*. John Wiley, 2nd edition, 1997.

[100] T. Kerr. Status of CR-like lower bounds for nonlinear filtering. *IEEE Trans. on Aerospace and Electronic Systems*, 25:590–601, 1989.

[101] A. Kong, J. S. Liu, and W. H. Wong. Sequential imputations and Bayesian missing data problems. *J. Amer. Stat. Assoc.*, 89(425):278–288, 1994.

[102] S.C. Kramer. *The Bayesian Approach to Recursive State Estimation: Implementation and Application*. PhD thesis, University of California, San Diego, 1985.

[103] S.C. Kramer and H.W. Sorenson. Bayesian parameter estimation. *IEEE Transactions on Automatic Control*, 33:217–222, 1988.

[104] S.C. Kramer and H.W. Sorenson. Recursive Bayesian estimation using piecewise constant approximations. *Automatica*, 24:789–801, 1988.

[105] V. Krishnamurthy and R.J. Elliott. Exact finite-dimensional filters of doubly stichastic auto-regressive processes. *IEEE Transactions on Automatic Control*, 42(9):1289–1293, 1997.

[106] R. Kulhavý. *Recursive Nonlinear Estimation – A Geometric Approach*. Lecture Notes in Control and Information Sciences. Springer-Verlag, 1996.

[107] T.D. Larsen. *Optimal Fusion of Sensors*. PhD thesis, Dept. of Automation, Technical University of Denmark, 1998.

[108] F. Le Gland, C. Musso, and N. Oudjane. An analysis of regularized interacting particle methods for nonlinear filtering. In *Proc. 3rd IEEE Workshop on Computer-Intensive Methods in Control and Data Processing*, Prauge, 1998.

[109] E.L. Lehmann. *Theory of point estimation.* Statistical/Probability series. Wadsworth & Brooks/Cole, 1991.

[110] J.S. Liu and R. Chen. Sequential Monte Carlo methods for dynamic systems. *J. Amer. Statist. Assoc.*, 93, 1998.

[111] J.S. Liu, R. Chen, and F. Liang. Monte Carlo filter for dynamic systems with application to target tracking in clutter. In *Proc. 3rd IEEE Workshop on Computer-Intensive Methods in Control and Data Processing*, Prague, 1998.

[112] A. Logothetis and V. Krishnamurthy. A Bayesian expectation–maximization framework for estimating jump Markov linear systems. *IEEE Transactions on Signal Processing*, 1999. Accepted for publication.

[113] Stéphane Mallat. *A Wavelet Tour of Signal Processing.* Academic Press, San Diego, 1998.

[114] MATLAB. *Language Reference Manual, Version 5.* The MathWorks, Inc., December 1996.

[115] D.Q. Mayne. A solution to the smoothing problem for linear dynamic systems. *Automatica*, 4:73–92, 1966.

[116] N. Metropolis, N. Rosenblutt, A.W. Rosenblutt, M.N. Teller, and E. Teller. Equations of state calculations by fast computing machines. *The Journal of Chemical Physics*, 21:1087–1091, 1953.

[117] R. M. Neal. Probabilistic inference using Markov chain Monte Carlo methods. Technical Report CRG-TR-93-1, Department of Computer Science, University of Toronto, 1993.

[118] H. Niederreiter. Quasi-Monte Carlo methods and pseudo-random numbers. *Bulletin of the American Mathematical Society*, 84:957–1041, 1978.

[119] J. Palmqvist. *On Integrity Monitoring of Integrated Navigation Systems.* Linköping Studies in Science and Technology. Thesis No 600, 1997.

[120] A. Papoulis. *Probability, Random Variables, and Stochastic Processes.* McGraw Hill, London, 1965.

[121] D.N. Politis. Computer-intensive methods in statistical analysis. *IEEE Signal Processing magazine*, 15(1), January 1998.

[122] W.K. Pratt. *Digital Image Processing.* John Wiley, 1978.

[123] G.W. Pulford and R.J. Evans. Probabilistic data association for systems with multiple simultaneous measurements. *Automatica*, 32(9):1311–1316, 1996.

[124] G.W. Pulford and A. Logothetis. An expectation-maximization tracker for multiple observations of a single target in clutter. In *Proc. 36th IEEE Conf. on Decision and Control*, 1997.

[125] L.R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proc. of the IEEE*, 1989.

[126] B.D. Ripley. *Stochastic Simulation*. John Wiley, 1988.

[127] C.P. Robert. *The Bayesian Choice: A Decision-Theoretic Motivation*. Springer Texts in Statistics. Springer Verlag, 1994.

[128] G.O. Roberts. Markov chain concepts related to sampling algorithms. In Gilks et al. [73].

[129] D.B. Rubin. Using the SIR algorithm to simulate posterior distributions. In J.M. Bernardo, M.H. DeGroot, D.V. Lindley, and A.F.M. Smith, editors, *Bayesian Statistics 3*, pages 395–402. Oxford University Press, 1988.

[130] L. Scharf. *Statistical Signal Processing: Detection, Estimation, and Time Series Analysis*. Addison-Wesley, 1991.

[131] C. Schneider. Structural inference and a modification of Dempster's combination rule. *Comp. Stat. & Data Analysis*, 10:331–338, 1990.

[132] G. Shafer. *A Mathematical Theory of Evidence*. Princeton University Press, Princeton and London, 1976.

[133] M. Simandl and J. Královec. Aspects of point-mass approach and grid design. In *3rd IEEE Workshop Computer Intensive–Methods in Control and Data Processing*, pages 57–63, 1998.

[134] M. Simandl, J. Královec, and P. Tichavský. Predictive and filtering lower bounds for nonlinear filters. In *IFAC World Congress*, Beijing, China, 1999.

[135] G.M. Siouris. *Aerospace Avionics Systems*. Academic Press, 1993.

[136] H.W. Sorenson, editor. *Kalman Filtering: Theory and Applications*. IEEE press, New York, 1985.

[137] H.W. Sorenson. Recursive estimation for nonlinear dynamic systems. In J. C. Spall, editor, *Bayesian Analysis of Time Series and Dynamic Models*, chapter 6. Marcel Dekker inc., 1988.

[138] H.W. Sorenson and D.L. Alspach. Recursive Bayesian estimation using Gaussian sum. *Automatica*, 7:465–479, 1971.

[139] P. Stoica and B. Ottersten. The evil of superefficiency. *Signal Processing*, 55: 133–136, 1996.

[140] M. Svensson. Implementering och utvärdering av Bayesiansk terrängnavigering. Masters Thesis LiTH-ISY-EX-2039, Department of Electrical Engineering, Linköping University, 1999. In swedish.

[141] W. Sweldens. *Construction and Applications of Wavelets in Numerical Analysis*. PhD thesis, Department of Computer Science, Katholieke Universiteit Leuven, Belgium, 1994.

[142] W. Sweldens and P. Schröder. Building your own wavelets at home. Technical Report IMI 1995:5, Dept. of Mathematics, University of South Carolina, 1995.

[143] H. Tanizaki. Nonlinear and nonnormal filters using monte carlo methods. *Comp. Stat. Data Anal.*, 25:417–439, 1997.

[144] J.H. Taylor. Cramer-Rao estimation error lower bound analysis for nonlinear systems with unknown deterministic variables. *IEEE Transactions on Automatic Control*, 24(2), 1979.

[145] The MCMC Preprint Service. `http://www.stats.bris.ac.uk/MCMC/`, 1999.

[146] P. Tichavský, C. Muravchik, and A. Nehorai. Posterior C-R bounds for performance of adaptive parameter tracking. In *Proc. 1st European Conf. on Signal Analysis and Prediction*, 1997.

[147] P. Tichavský, C. Muravchik, and A. Nehorai. Posterior Cramér-Rao bounds for discrete-time nonlinear filtering. *IEEE Transactions on Signal Processing*, 46(5):1386–1396, 1998.

[148] L. Tierney. Markov chains for exploring posterior distributions. *Ann. Stat.*, 22(4):1701–1762, 1994.

[149] L. Tierney. Introduction to general state-space Markov chain theory. In Gilks et al. [73].

[150] P.J. Van Laarhoven and E.H.L. Aarts. *Simulated Annealing: Theory and Application*. Reidel Pub., Amsterdam, 1987.

[151] H.L. Van Trees. *Detection, Estimation and Modulation Theory*. Wiley, New York, 1968.

[152] P.K. Varshney. *Distributed detection and data fusion*. Springer, New York, 1997.

[153] M. Vidyasagar. Statistical learning theory and randomized algorithms for control. *IEEE Control Systems Magazine*, 18(6), December 1998.

[154] E. Waltz and J. Llinas. *Multisensor data fusion.* Artech House, Boston, 1990.

[155] H.M. Youssef. Spline filter for optimum phase demodulation. *Stochastics*, 2: 105–137, 1976.

Notation

Common abbreviations, and general notational conventions for mathematical symbols and operands follow. When local differences occur, they are clearly indicated in the text.

Symbols

$\boldsymbol{x}$	Bold faced letters are stochastic variables. No notational distinction is used for vectors and scalars.
$\hat{x}$, $\hat{x}(y)$	Sample estimate of the stochastic variable $\boldsymbol{x}$ based in the observation that $\boldsymbol{y} = y$.
$\hat{x}(\boldsymbol{y})$	Estimator of $\boldsymbol{x}$, thus a stochastic variable itself.
$\boldsymbol{x}_t$	Stochastic discrete time process.
$\mathbb{X}_t$	Stacked vector of the process x_t from initial time until time t.
$\mathbb{X}_{s:t}$	Stacked vector $\mathbb{X}_{s:t}^T = [x_s^T, x_{s+1}^T, \ldots, x_t^T]$.
$p(x)$	Density for the stochastic variable $\boldsymbol{x}$.
$p_x(z)$	Density for $\boldsymbol{x}$, evaluated at z.
$\mathrm{N}(x; m, P)$	Gaussian probability density function with mean vector m and covariance matrix $P > 0$.

$\mathcal{U}(a,b)$	Uniform distribution on the interval $[a,b]$.
$\mathcal{N}(m,P)$	Gaussian distribution with mean vector m and covariance $P \geq 0$.
$p(x,y)$	Joint density for $\boldsymbol{x}$ and $\boldsymbol{y}$.
$p(x \mid y)$	Density for the random variable $\boldsymbol{x}$ given that the random variable $\boldsymbol{y} = y$.
$\mathbb{R}^n$	Euclidean n-dimensional space.
$\mathbb{N}$	The set of natural numbers, i.e., positive integer numbers.

Operators and Functions

$\|v\|_Q$	Weighted norm of matrix or vector, $\|v\|_Q^2 = v^T Q v$, whenever the multiplications make sense.
$\|v\|$	Unitary weight matrix, $\|v\|^2 = v^T v$.
A^T	Transpose of the matrix A.
A^{-1}	Inverse of the matrix A.
$\operatorname{tr} A$	Trace of the matrix A, i.e., the sum of the diagonal elements of A.
$\|A\|$	Determinant of the matrix A.
$A^\dagger$	Moore-Penrose pseudo inverse $A^\dagger = (A^T A)^{-1} A^T$.
$A > 0$	The symmetric matrix A is positive definite.
$A \geq 0$	Positive semidefinite matrix A.
$\mathsf{E}\, x$	Expectation of the random variable $\boldsymbol{x}$.
$\mathsf{Cov}\, x$	Covariance matrix of the random variable $\boldsymbol{x}$.
∇_x	Gradient operator, column vector of partial derivatives w.r.t. the entries in x.
Δ_y^x	Laplacian operator, $\Delta_y^x = \nabla_y \nabla_x^T$.
$\arg\max_x$	The argument x that maximizes operand.

Abbreviations

TAN	Terrain Aided Navigation.
DTED	Digital Terrain Elevation Database.
INS	Inertial Navigation System.
GPS	Global Positioning System.
DGPS	Differential GPS.
i.i.d.	Independent Identically Distributed
MMSE	Minimum Mean Square Error.
MV	Minimum Variance.
RMS	Root Mean Square.
MAP	Maximum A Posteriori.
KF	Kalman Filter
EKF	Extended Kalman Filter.

PMF	Point-Mass Filter.
FIM	Fisher Information Matrix.
CEP	Circular Error Probable.
MC	Monte Carlo.
MCMC	Markov Chain Monte Carlo.
SIS	Sequential Importance Sampling.
SIR	Sampling Importance Resampling.
EM	Expectation Maximization.
MSMF	Multiple Simultaneous Measurement Filter.
EMDA	Expectation Maximization Data Association.
MCDA	Monte Carlo Data Association.
ML	Maximum Likelihood

Index

PhD Dissertations, Division of Automatic Control, Linköping University

M. Millnert: Identification and control of systems subject to abrupt changes. Thesis no. 82, 1982. ISBN 91-7372-542-0.

A.J.M. van Overbeek: On-line structure selection for the identification of multivariable systems. Thesis no. 86, 1982. ISBN 91-7372-586-2.

B. Bengtsson: On some control problems for queues. Thesis no. 87, 1982. ISBN 91-7372-593-5.

S. Ljung: Fast algorithms for integral equations and least squares identification problems. Thesis no. 93, 1983. ISBN 91-7372-641-9.

H. Jonson: A Newton method for solving non-linear optimal control problems with general constraints. Thesis no. 104, 1983. ISBN 91-7372-718-0.

E. Trulsson: Adaptive control based on explicit criterion minimization. Thesis no. 106, 1983. ISBN 91-7372-728-8.

K. Nordström: Uncertainty, robustness and sensitivity reduction in the design of single input control systems. Thesis no. 162, 1987. ISBN 91-7870-170-8.

B. Wahlberg: On the identification and approximation of linear systems. Thesis no. 163, 1987. ISBN 91-7870-175-9.

S. Gunnarsson: Frequency domain aspects of modeling and control in adaptive systems. Thesis no. 194, 1988. ISBN 91-7870-380-8.

A. Isaksson: On system identification in one and two dimensions with signal processing applications. Thesis no. 196, 1988. ISBN 91-7870-383-2.

M. Viberg: Subspace fitting concepts in sensor array processing. Thesis no. 217, 1989. ISBN 91-7870-529-0.

K. Forsman: Constructive commutative algebra in nonlinear control theory. Thesis no. 261, 1991. ISBN 91-7870-827-3.

F. Gustafsson: Estimation of discrete parameters in linear systems. Thesis no. 271, 1992. ISBN 91-7870-876-1.

P. Nagy: Tools for knowledge-based signal processing with applications to system identification. Thesis no. 280, 1992. ISBN 91-7870-962-8.

T. Svensson: Mathematical tools and software for analysis and design of nonlinear control systems. Thesis no. 285, 1992. ISBN 91-7870-989-X.

S. Andersson: On dimension reduction in sensor array signal processing. Thesis no. 290, 1992. ISBN 91-7871-015-4.

H. Hjalmarsson: Aspects on incomplete modeling in system identification. Thesis no. 298, 1993. ISBN 91-7871-070-7.

I. Klein: Automatic synthesis of sequential control schemes. Thesis no. 305, 1993. ISBN 91-7871-090-1.

J.-E. Strömberg: A mode switching modelling philosophy. Thesis no. 353, 1994. ISBN 91-7871-430-3.

K. Wang Chen: Transformation and symbolic calculations in filtering and control. Thesis no. 361, 1994. ISBN 91-7871-467-2.

T. McKelvey: Identification of state-space models from time and frequency data. Thesis no. 380, 1995. ISBN 91-7871-531-8.

J. Sjöberg: Non-linear system identification with neural networks. Thesis no. 381, 1995. ISBN 91-7871-534-2.

R. Germundsson: Symbolic systems – theory, computation and applications. Thesis no. 389, 1995. ISBN 91-7871-578-4.

P. Pucar: Modeling and segmentation using multiple models. Thesis no. 405, 1995. ISBN 91-7871-627-6.

H. Fortell: Algebraic approaches to normal forms and zero dynamics. Thesis no. 407, 1995. ISBN 91-7871-629-2.

A. Helmersson: Methods for robust gain scheduling. Thesis no. 406, 1995. ISBN 91-7871-628-4.

P. Lindskog: Methods, algorithms and tools for system identification based on prior knowledge. Thesis no. 436, 1996. ISBN 91-7871-424-8.

J. Gunnarsson: Symbolic methods and tools for discrete event dynamic systems. Thesis no. 477, 1997. ISBN 91-7871-917-8.

M. Jirstrand: Constructive methods for inequality constraints in control. Thesis no. 527, 1998. ISBN 91-7219-187-2.

U. Forssell: Closed-loop identification: methods, theory, and applications. Thesis no. 566, 1999. ISBN 91-7219-432-4.

A. Stenman: Models on demand: algorithms, analysis and applications. Thesis no. 571, 1999. ISBN 91-7219-450-2.