Fundamentals of Sensor Fusion

Target tracking, navigation and SLAM

Edmund Brekke

HTTPS://WWW.NTNU.EDU/EMPLOYEES/EDMUND.BREKKE

First edition, August 31, 2019

Contents

1. Overview of tracking, navigation and SLAM

Wikipedia defines sensor fusion as *combining of sensory data or data derived from disparate sources such that the resulting information has less uncertainty than would be possible when these sources were used individually*. Depending on the application, this entails different levels of interpretation of the data. Sensor fusion is closely related to *state estimation*, as the goal is typically to estimate a state vector for an underlying dynamical system based on noisy measurements.

In the context of state estimation, there are two fundamental approaches to sensor fusion. On the one hand, it is possible to process measurements from the different sensors as they arrive, updating the state estimate every time this happens. This is known as *measurement level fusion*. On the other hand, it is possible to maintain different estimates for the different sensors, and then fuse them into a combined estimate. This latter approach is in the context of target tracking known as *track-to-track fusion*. In this book, the main focus will be on the former approach. Therefore, this is not really a book about sensor fusion, but rather about state estimation. However, the estimation problems that dominate in typical sensor fusion applications have many things in common which distinguish them from other estimation problems, and and a solid understanding of applied state estimation is essential as a fundament for more advanced work in sensor fusion.

1.1 Target tracking, navigation and SLAM

In this book the focus will be on three sensor fusion applications: Target tracking, inertial navigation and simultaneous localization and mapping (SLAM). In target tracking the goal is to estimate the motion of one or several moving objects by means of sensors that observe these objects. Such sensors can include cameras, radar, laser scanners and sonar. In inertial navigation the goal is to estimate the motion of a moving object by means of sensors attached to that object. This includes inertial sensors such as an accelerometer and a gyroscope, and sensors such as GNSS and compass which measure location and orientation relative to an external coordinate system. In SLAM the goal is again to estimate own motion as in inertial navigation, but it is also a goal to build a map of the environment. For this reason, SLAM must also use imaging sensors such as those used in target tracking.

We shall refer to the sensors used in target tracking as *exteroceptive*, while we shall refer

to the sensors used in inertial navigation as *interoceptive*. We may also refer to exteroceptive sensors as imaging sensors. While it perhaps is an abuse of terminology to describe GPS as interoceptive, this terminology makes it easy to distinguish the two categories of sensors. The processing of interoceptive and exteroceptive data is radically different. Interoceptive data (e.g., a GPS measurement) can be fed directly to a Kalman filter, which then will estimate position and other kinematic quantities[1]. For exteroceptive data, which come as images or scans, we must first extract relevant features by means of detection and segmentation procedures. This will typically give a list of blobs, which can be reduced to point features by means of clustering procedures. To be able to estimate anything from these features, it is necessary to establish correspondences between features in consecutive scans. This is known as *data association*.

Consequently, the processing pipeline in target tracking and SLAM is considerably more complex than in inertial navigation. but it is also important to understand that solutions to the different applications are dictated by rather different requirements. Inertial navigation may seem like the simplest problem since it does not involve data association, but the requirements of accuracy and robustness are generally very strict. Therefore inertial navigation uses more sophisticated models, and clever choices of estimator design are needed. On the other hand, target tracking tends to use extremely simple motion models, since it would be futile and even dangerous to attempt to estimate more complex models. While a typical navigation system uses 16 or more dimensions in the state vector, most tracking methods use only 4 or 5 dimensions in the state vector.

In inertial navigation it is often desired to make solutions with global stability properties. Such properties provide a guarantee that the filter will not diverge. For target tracking, where all reasonable motion models are marginally stable, it is impossible to guarantee stability. Indeed, if the system is allowed to run for long enough time, divergence is bound to happen. The best safeguard against this is ensure that all plausible hypotheses for data association are considered. For this reason, the computational complexity of target tracking will typically be substantially higher than for inertial navigation.

For SLAM, data association is also an issue that must be taken seriously, but it is less critical than for target tracking. The reason for this is that SLAM typically uses hundreds and thousands of landmarks in the environment. Even if the method fails to associate a large percentage of these, the remaining landmarks are still likely to carry enough information to support estimation of own motion. However, the large number of landmarks is also something that drives computational complexity up. A key research topic in SLAM has therefore been to exploit structure inherent in the SLAM problem to enable fast state estimation.

1.2 A brief history of sensor fusion

Research in target tracking has historically been driven by military applications. The radar was developed just before and during the early phases of World War 2. Most of the tracking algorithms currently in use were developed during the cold war. These were typically designed for guiding antiaircraft artillery or antisubmarine torpedoes. Variations of these methods have dissipated into virtually any defence application where the goal is to shoot someone or something.

Target tracking found its first civilian applications in air traffic control (ATC) and as a maritime navigation aid, where it is known as automatic radar plotting aid (ARPA). The main purpose of these systems is to enable human operators to prevent collisions, by keeping track of the whereabouts of other airplanes and ships, as well as unwelcome intruders such as drones. These systems have undergone progressive steps towards autonomy. Since the Überlingen aircraft collision in 2002 pilots have been instructed to trust their automatic collision avoidance system over human instructions.

[1] Wild-point filtering should of course be used to prevent obviously erroneous measurements to do any damage.

Target tracking also plays a role in all kind of surveillance applications. This can be surveillance of secured areas where one wants to know if unwelcome intruders are passing through or loitering. It can be biological or industrial applications where one wants to follow the motion of fish, bacteria, insects, etc. Tracking methods can also be used to keep track of space debris that poses a threat to satellites or astronauts.

In the same way as for target tracking, early applications of inertial navigation also emerged in World War 2 due to military needs. Long-range missiles such as the German V2 rockets needed a navigation system to hit their targets. The development of inertial navigation continued with the more advanced missiles developed during the cold war. At the same time, the space race between the United States and the Soviet Union also led to the usage of advanced navigation technology in spacecraft and satellites. The moonlanding is often credited as the first application of the Kalman filter. Since then, inertial navigation has become commonplace technology in all kinds of vehicles, and also electrical gadgets such as smartphones. The next stepping stone was the introduction of Global Navigation Satellite Systems (GNSS), including the Global Positioning System (GPS), which since 1995 has since then provided a global reference frame for navigation systems, so that location anywhere on the earth can be estimated with an accuracy of a handful of meters, or possibly centimeters if differential techniques are used.

However, access to satellites is limited in many situations, such as for underwater vehicles or indoor robotics. This limitation has since the late 90s been a major driving force for the development of SLAM. Using SLAM, a robot can estimate its own location relative to its local environments, such as the rooms in a building. SLAM also has roots in computer vision, virtual reality and 3-D reconstruction.

The primary motivation behind the course TTK4250 Sensor Fusion is current research on autonomous vehicles. Since the DARPA Grand Challenge was won by the driverless car Stanley in 2005 it has been clear that autonomous cars can successfully cope with fairly complex environments, and the automotive industry has since then been putting huge efforts into the development of autonomous cars. Both target tracking, inertial navigation and SLAM are essential components of the software for autonomous cars. Norway does not have a large automotive industry, but its maritime industry is world-leading. Increased levels of autonomy are expected to have a huge impact on this industry in the near future.

A secondary motivation behind TTK4250 Sensor Fusion is that estimation in general plays an increasing role both in cybernetics and related disciplines. A general trend is that as simpler problems are being solved, researchers and innovators move on to tackle more challenging problems. The simpler problems are in this context those that can be solved by feedback control alone, possibly including some filtering or observers to process sensor data. The more challenging problems are those that involve a more complex interpretation of the data, and where the control system consequently must perform some kind of reasoning in the presence of uncertainty. The fundamental tools such as Kalman filters, particle filters, Gaussian mixtures and graphical models appear again and again in such applications. The more general management of uncertainty is highly relevant in several other fields, including subsurface imaging for oil and gas exploration, medical imaging, meterology and climate science, system identification of physical and biological processes and financial markets, to mention a few areas.

1.3 Modeling - important or not so important?

Imagine that you hear the roar of an F16 somewhere to the west. Soon after you hear it right above you. Instinctively, you then try to spot it to your east, where you already can see it fly into the horizon. In this process, you utilize several models. First, you have a *plant model*, also known as process model or kinematic prior, which tells you that the plane probably continues more or less in the same direction as it moved earlier. Second, you have a *measurement model*, which in this

particular example would account for the fact that the airplane probably was in the direction from which the sound was emitted when the sound was emitted, and which also would account for the fact that the speed of sound is not that much higher than the speed of the airplane.

It is fairly evident that models have a central role to play in sensor fusion. For the choice of models, the Einstein maxim applies: "As simple as possible, but not simpler". Theoretically, we could make a model of the F16 plane which accounts for its motion in 6 degrees of freedom, including various dynamical forces. Unfortunately, there is no way that we can estimate all the state variables of such a model. Not only would it be meaningless to use such a model, but it would most certainly be quite disastrous, as the inevitable failures to estimate some of the states very soon will cause the track to diverge significantly from the truth. In contrast, most tracking methods use models assuming nearly constant velocity (the CV model) or nearly constant turn rate (the CT model).

The modeling task does not necessarily stop with such kinematic models, e.g., the kind of models that are used in a Kalman filter. If you see several airplanes instead of only one, then you are forced to evaluate which of these is that particular F16 plane. If you do not see the plane again, perhaps you will consider the possibility that the roar was from a meteorite fall from the sky instead. However, you are in possession of some vague prior knowledge about how frequently F16s and meteorites fly above your head, and this knowledge would probably make you extremely reluctant to accept this alternative explanation, and you would still search for the F16 in the sky. Especially in target tracking, this kind of knowledge is often encoded in models for target existence, false alarms and misdetections. Even if all these models on their own are very simple, the overall tracking problem, where all the models are combined, will quickly become rather complex.

It is important that models make sense and exhibit sufficient degree of realism, especially when expanding modeling from basic kinematic properties to more exotic properties. Sometimes one may discover that an estimation method that in theory should work extremely well, fails in any realistic scenario because its modeling assumptions are entirely unrealistic. Also, adapting a tested and tried method to a new problem may fail to be successful if that problem has characteristics which violate the assumptions of that method.

Machine learning methods are often viewed as an alternative to model-based methods. This is a valid point of view, although with two important caveats. It cannot be emphasized strongly enough that the term machine learning is a wide umbrella that covers several loosely related methods. As the first caveat, so-called probabilistic machine learning methods do involve explicit models, which are to be learned from the data. As the second caveat, artificial neural networks are fundamentally nonlinear regression engines which encode relationships between inputs and outputs in a black box fashion. In principle there are no limits to the relationships that such an engine can learn. It could for example learn the relationship between GPS and IMU measurements and the true state vector in a navigation system. But this relationship can be viewed as an implicit representation of the kinematic model. The question then is whether the learned model will give better predictive power than a model specified manually from prior knowledge. For complex problems such as image recognition this is probably the case, while it is much more questionable for, e.g., inertial navigation using GPS and IMU.

1.4 The deterministic and probabilistic paradigms

Uncertainty can be dealt with in many ways. First of all, we may ignore it alltogether. Many classical estimation techniques exist which do not actually utilize models of uncertainty. In standard unweighted least squares estimation one simply aims to find the underlying parameter that minimizes the average error that results between observations and predicted observations. As another example, to match two images one may extract some features in both images, and attempt to find a transformation that makes as many of the features as possible coincide.

State estimation is often referred to as filtering, and state estimators are often known as filters (e.g., the Kalman filter). Filtering is a fairly general concept. One can design filters which do not correspond to a meaningful estimators, and one can design filters that estimate a state without any explicit treatment of uncertainty. Instead, a filter is typically designed by specifying requirements on the spectral properties of the estimation error. For example, in dynamical positioning the goal is to estimate long term motion of a vessel, while short term disturbances due to sea waves should be ignored. This way of thinking leads naturally to the theory of linear and nonlinear observers, whose tuning parameters (gains) are tuned according to the desired response time of the observer. The designer of the observer will typically present a stability proof to convince that the observer can be relied upon.

The conventional approach in sensor fusion is, however, to quantify uncertainty by means of probabilities. This is quite natural, as probability has been the established language of uncertainty since the time of Laplace. There are several reasons why we would like to quantify uncertainty, as opposed to merely coping with uncertainty. First of all, established estimation algorithms such as the Kalman filter have been developed in a probabilistic context. Furthermore, quantifying the uncertainty is useful to help us with interpreting the results. Returning to the F16 example, having an uncertainty estimate makes it possible for you to decide how much you should move your gaze around the spot where you expect to see it. As a third reason, quantified uncertainty enables us to make precise rules for how to weigh different pieces of evidence against each other in a (multi-) sensor fusion system.

Such rules may be perceived as a straightjacket or as the beams of a house construction. This book will attempt to convince you towards the latter perception. It is extremely important to have sensible structures to rely upon when designing complex sensor fusion systems. At the core of these structures we often find Bayes' rule, which is a very powerful tool, because it governs how information changes in the presence of new evidence. Proving that a sensor fusion method obeys Bayes' rule is in some sense the equivalent of stability proofs in control theory. The virtue of a stability proof is not necessarily that the system will converge to zero error. If noise continuously enter the system, that will not happen. The stability proof may, however, ensure that the system will not do anything crazy. Similarly, if a sensor fusion method is faithful to Bayes, we can trust it to behave in a rational manner.

1.5 References and rationale

Several excellent books on estimation and sensor fusion exist. Why was it then necessary to write a textbook from scratch for the course TTK4250? The answer has to do with the limitations of how much knowledge a student can be expected to digest during a 7.5 study points course. In TTK4250 the goal is to reach sufficient comprehension of state-of-the-art methods for tracking, navigation and SLAM to be able to use these from day one in research problems in the 5th year specialization projects and MSc theses. Furthermore, this goal is to be achieved from a foundation consisting of little more than a basic probability course and a basic Kalman filtering and estimation course (i.e., linear systems theory). There exists no textbook today that covers this entire span. Nevertheless, I am convinced that it is possible to cover this in a one-semester course, if there is a strong focus on the core methods and their theoretical foundations, and less focus on general theory or the many possible variations of the methods.

If this course was to use an existing textbook, the most natural candidates would have been the 2001 book by Bar-Shalom, Li & Kirubarajan [3] or the 2012 book by Gustafsson [42]. Both cover the fundamentals of probability and estimation extremely well, and include authoritative in-depth treatments of Kalman filtering, several nonlinear methods (such as EKF) and multiple model estimation. Any student who wants to do serious work in sensor fusion should own a copy of at least one of these. There are, however, two reasons why I have chosen not to use any of these as

curriculum for TTK4250. Both books are very rich on details, and I am concerned that the amount of details will simply be too much for many students to digest. Furthermore, neither book covers data association, which in my opinion is one of the most important topics in sensor fusion.

Fundamental probability and estimation theory is covered in Chapters 2 and 3, but only to a limited extent. For more details the reader is referred to Kay's book [51] from 1993 and to Papoulis & Pillai's textbook on probability and stochastic processes from 2002 [70]. Again, any serious sensor fusion student should be familiar with these books. While [51] is very readable and takes a discrete-time approach, [70] is the definitive reference on the theory of continuous-time stochastic processes.

For target tracking, I am first and foremost inspired by Bar-Shalom and Li, represented by the book [2], which was written in 1995. A more expanded version of this book came in 2011 with the title title "Tracking and Data Fusion: A Handbook of Algorithms" [5]. The most comprehensive textbook on target tracking that exists is probably [10]. It is indispensable for diving into the building blocks of MHT implementations, but cannot be used as an authoritative source on the state-of-the-art with its 20 years of age. To become familiar with the modern paradigm in target tracking the reader would have to supplement the mentioned books with Challa [20] and Mahler [59] (or the research articles that these books build upon). In [20] methods that use the concept of existence probability are described. The book is to an even larger extent than [5] written as a handbook of algorithms. In [59], the focus is on Mahler's theory of random finite sets. While Mahler has made significant efforts to present this material in a pedagogical manner, the material itself is far too demanding to be used in a basic course on sensor fusion.

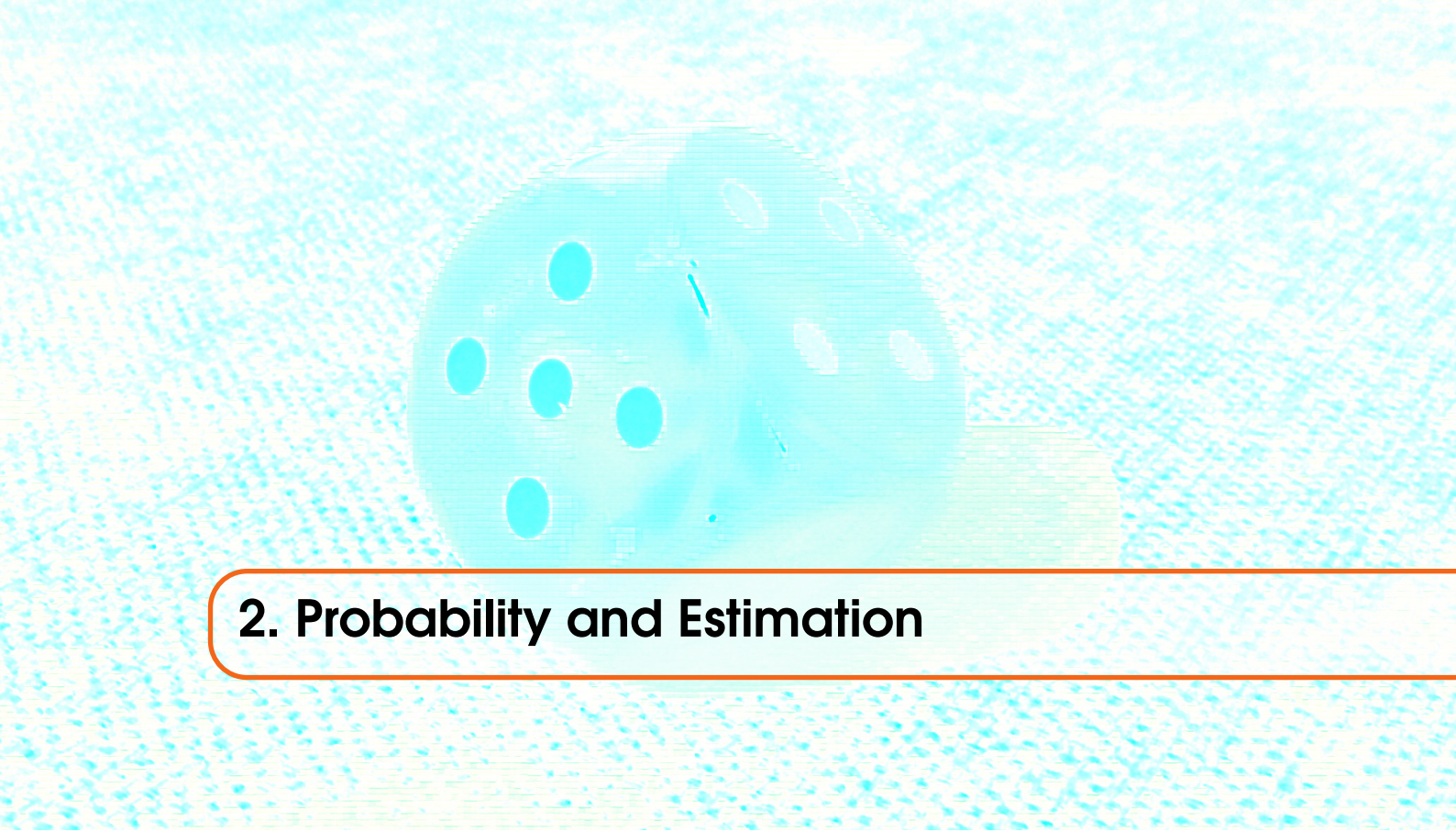
As for navigation, established textbooks include the books by Groves [40] and Titterton & Weston [85]. These delve much deeper into the fundamentals of GNSS technology etc. than we would like to do in this course, while their exposition of the error state formulation of the Kalman filter is much less systematic. Inertial navigation has previously been taught in the course TTK5 using Vik's book [89]. Again, the main shortcoming is no systematic exposition of the error state formulation. This is, however, available in Sola's text [81], which forms the basis for our treatment of inertial navigation.

There exists no textbook on SLAM that is up to date with the significant developments that have found place during the last 10 years. The closest one gets to a canonical textbook on SLAM remains Thrun's book [84] which is from 2006. It gives a very detailed treatment of EKF-SLAM and particle filtering SLAM. Recursive SLAM methods, together with the related topic of localization, are also covered in [42]. While not primarily about SLAM, Barfoot's book [6] may be very relevant for anyone working on SLAM and navigation.

1.6 Acknowledgement

This book is largely inspired by my experience in supervising MSc and PhD students working on autonomous ship projects during the last 5 years. All of these students can be found on the website `folk.ntnu.no/edmundfo/autoseastudents/autoseastudents.html`. In particular, I would like to express special appreciation to Lars-Christian Tokle, who has contributed ideas, proofreading and some of the central proofs, and Michael Ernesto Lopez who has made several of the TiKZ-coded figures. The book has been written using Texmaker and the Legrand Orange Book template (`www.latextemplates.com/template/the-legrand-orange-book`). During the writing I upgraded my macbook to a 2018 model with butterfly keys. I blame all typos on the butterfly keys.

Finally the reader should be aware that this is still work in progress. As of 14th of August, only the first three chapters are completed and will be released on Blackboard. Additional chapters will be released as soon as they are completed.



2. Probability and Estimation

2.1 Foundations of probability theory

Probability theory is about assigning numbers to *events*. These numbers, known as probabilities, tell us something about how likely these events are to happen. This intuitive notion of probability can be expressed more precisely in terms of three axioms. Let the letter E refer to any event, and let Ω be the outcome space, that is, the collection of all possible events. We use the notation $\Pr\{\cdot\}$ to signify probability of the events. The axioms of probability can then be formulated as follows

1. $\Pr\{E\} \in \mathbb{R}$, $\Pr\{E\} \geq 0$.
2. $\Pr\{\Omega\} = 1$.
3. For any sequence of disjoint events $E_1, \ldots, E_n$ we have $\Pr\{\bigcup_{i=1}^{n} E_i\} = \sum_{i=1}^{n} \Pr\{E_i\}$.

The first axiom then tells us that the probability of any event is a non-negative real number. The second axiom tells us that the probability of the entire outcome space is one. The third axiom tells us that the probabilities of disjoint events can be added in order to find the probability that any of the events considered should happen.

In addition to the axioms, probability theory also relies on some fundamental definitions, which provide a language for how different events can be related to each other.

Definition 2.1.1 — Conditional probability. Given two events A and B, the conditional probability of A given B is

$$\Pr\{A|B\} = \frac{\Pr\{A \cap B\}}{\Pr\{B\}} \tag{2.1}$$

Definition 2.1.2 — Independence. We say that two events A and B are independent if

$$\Pr\{A \cap B\} = \Pr\{A\}\Pr\{B\} \tag{2.2}$$

Equipped with these definitions, we can establish some basic results in probability theory. The following two results will play a key role in very many of the derivations in the subsequent chapters.

Theorem 2.1.1 — The total probability theorem. Let $\{B_n\} = \{B_1, B_2, B_3, \ldots\}$ be a countable partition of the outcome space, and let A be some event. Then the following is true:

$$\Pr\{A\} = \sum_n \Pr\{A|B_n\}\Pr\{B_n\}.$$

Proof. The probability of A is the same as the probability of the union $\bigcup_n (A \cap B_n)$ insofar as $\{B_n\}$ is a partition of the outcome space. Thus

$$\Pr\{A\} = \sum_n \Pr\{A \cap B_n\} = \sum_n \Pr\{A|B_n\}\Pr\{B_n\}$$

where the first equality follows from axiom number 3 and the last equality follows from the definition of conditional probability. ■

Theorem 2.1.2 — Bayes' rule. For any two events A and B, the following is true:

$$\Pr\{A|B\} = \frac{\Pr\{B|A\}\Pr\{A\}}{\Pr\{B\}}.$$

Proof. According to the definition of conditional probability the joint probability of A and B is $\Pr\{A \cap B\} = \Pr\{A|B\}\Pr\{B\} = \Pr\{B|A\}\Pr\{A\}$. Bayes' rule follows from dividing by $\Pr\{B\}$ on both sides. ■

2.2 Random variables and probability distributions

It would be very cumbersome if we always were to work explicitly with events. In probability theory and its applications we are typically interested in the behavior of quantities that are of a random nature. By the phrase "of a random nature" we mean that the quantity is uncertain, and that our knowledge about it is modeled by means of probability theory. Such a quantity could for example be the speed of an airplane. In a probabilistic model we would have some probability that it is smaller than $100\,\mathrm{m/s}$, while the probability that it is larger than or equal to $100\,\mathrm{m/s}$ would be one minus that probability. Statements such as "the velocity is less than $100\,\mathrm{m/s}$" correspond to the events in Section 2.1.

2.2.1 Random variables and probability measures

Random variables are in the mathematical approach to probability theory defined as functions from an abstract outcome space to a more concrete outcome space such as $\mathbb{R}^n$. For any element in the abstract space the random variable attains a certain element in the concrete space, known as the *realization* of the random variable. The formal definition of a random variable is based on measure theory, which is one of the core foundations of mathematical analysis (other foundations being algebra and topology). The conceptual framework revolves around the concept of a *probability space*, which again builds on the concepts of σ*-algebras* and *measures*.

Definition 2.2.1 — σ-algebra. A σ-algebra $\mathscr{F}$ on a set Ω is a collection of subsets of Ω such that

- Ω itself is a member of $\mathscr{F}$.
- If the subset $A \subseteq \Omega$ is a member of $\mathscr{F}$, then its complement $\Omega \backslash A$ is also a member of $\mathscr{F}$.
- If A_1, A_2, A_3, etc are members of $\mathscr{F}$, then the union $\bigcup_n A_n$ is also a member of $\mathscr{F}$.

In probability theory, the elements of the σ-algebra plays the role of meaningful events. We can illustrate this with a simple example.

■ **Example 2.1 — Throw of a dice.** The outcome space when we throw a dice is $\Omega = \{1,2,3,4,5,6\}$. All the possible subsets of these 6 elements constitute a corresponding σ-algebra:

$$\mathscr{F} = \{\{1\},\{2\},\ldots,\{1,2\},\ldots,\{1,2,3,4,5,6\}\}.$$

We see that for any outcome at least one, and possibly several members of the σ-algebra will be active. ■

Measures are functions from σ-algebras to the real numbers. In other words, a measure is a systematic way of assigning numbers to subsets. In probability theory we are only concerned with a special class of measures known as probability measures.

Definition 2.2.2 — Probability measure. A probability measure P on the σ-algebra $\mathscr{F}$ is a function from $\mathscr{F}$ to the unit interval $[0,1]$ that obeys the three axioms of probability.

We see that the probability measure by definition obeys the fundamental axioms of probability that we introduced in the beginning of Section 2.1. We refer to the combination of the outcome space, the σ-algebra, and the probability measure as a probability space.

Definition 2.2.3 — Probability space. A probability space is a triplet $(\Omega,\mathscr{F},P)$ where

- Ω is a space of possible events.
- $\mathscr{F}$ is a σ-algebra on Ω.
- $P : \mathscr{F} \to [0,1]$ is a probability measure.

Definition 2.2.4 — Random variable. A random variable X is a function from Ω into another space $\mathbb{O}$, which we henceforth shall know as the outcome space.

The outcome space $\mathbb{O}$ can be many different things. In game of dice it will consist of the possible faces of a dice, which can be represented as $\{1,2,\ldots,6\}$. Most typically, $\mathbb{O}$ will be the space of real numbers $\mathbb{R}$, or it may the n-dimensional space of real-valued vectors $\mathbb{R}^n$. In the latter case, we often talk about random vectors instead of random variables. The random variable X connects the abstract space with the outcome space $\mathbb{O}$, so that probability becomes distributed on $\mathbb{O}$. The probability that the value of X is a member of B, where $B \subseteq \mathbb{O}$, is given by $P(X^{-1}(B))$, i.e., by summing the probabilities of all elements $\omega \in \Omega$ for which $X(\omega)$ ends up in B.

Why is it useful to introduce the abstract space Ω in addition to the more tangible outcome space $\mathbb{O}$? At least two answers can be provided. First, the definition of random variables as functions provides a clear mechanism for distinguishing random variables from their realizations. While the difference is intuitively easy to grasp, it is also nice to have a precise mathematical distinction. Second, the tangible outcome space $\mathbb{O}$ is not necessarily as neat and tidy as $\mathbb{R}^n$. For example, we shall in Chapter 10 let $\mathbb{O}$ consist of all finite subsets of $\mathbb{R}^n$. As another example, $\mathbb{O}$ could be a closed manifold such as $SO(3)$ or the surface of the earth. In such cases it may not be obvious how probability mass should be distributed in $\mathbb{O}$. By defining the random variables as functions we get a mechanism for mapping the probability mass from a space where probabilities by definition make sense to this more exotic space.

Definition 2.2.5 — Realization. The output of the random variable X for a particular $\omega \in \Omega$ is called a realization of X. We can write this as $x = X(\omega)$.

The concepts of a random variable and its realization are often confused, accidentally or deliberately. Often the same notation is used for both. This is often the only sensible thing to do, as notation would quickly become very complicated if we were to distinguish random variables and their realizations all the time. Nevertheless, the distinction should be remembered, as it sometimes can be quite important.

The abstract measure $P(\cdot)$ induces a probability measure on the tangible outcome space $\mathbb{O}$. We can relate this measure to probabilities as follows:

Definition 2.2.6 — Probability measure of a random variable. The probability measure of X is a function $\beta_X(\cdot)$ from subsets of $\mathbb{O}$ to $[0,1]$ so that

$$\beta_X(S) = \Pr\{X \in S\} = P(X^{-1}(S)). \tag{2.3}$$

2.2.2 Probability distributions

The main issue with probability measures, and the equivalent formulation in terms of primitive events in Section 2.1, is that these representations are highly redundant. To see this, and make the conceptual leap over to more useful representations, let us look at scalar continuous-valued random variables, i.e., let $\mathbb{O} = \mathbb{R}$. We do not need to assign a probability value to all possible subsets of $\mathbb{R}$ to specify the random properties of the random variable X. It would suffice to assign a probability value to all subsets of the form $(-\infty, x)$ where x is a real number, i.e., a possible realization of X. Or it would suffice to define a function whose integrals from $-\infty$ to x yields these values. These two viewpoints lead to the concept of cumulative distribution functions and probability density functions.

Definition 2.2.7 — Cumulative distribution function (cdf). The cdf, denoted $P(x)$, of $X \in \mathbb{R}$ is the probability $\Pr\{X < x\}$.

Definition 2.2.8 — Probability density function (pdf). The pdf of the scalar random variable X is the derivative

$$p(x) = \frac{\partial P(x)}{\partial x}.$$

Generally, the term probability distribution is used to mean the same as a pdf. It must be emphasized that *pdf's and probabilities are two different things.* We get probabilities when we integrate a pdf over a subset of the outcome space.

For discrete-valued random variables (e.g., throwing a dice) we do not normally talk about the pdf, but rather about the point mass function (pmf), which is given by $p(x) = \Pr\{X = x\}$. We can obtain the cdf in the same manner as for continuous-valued random variables by replacing integration with summation. Notice that such expressions such as $\Pr\{X = x\}$ in general are meaningless for continuous-valued random variables: The probability of X attaining a particular value will be zero if X is distributed over a continuous space.

The extension of cdfs and pdfs to vector-valued random variables is fairly straightforward. For example, if $Z = [X,Y]^\mathsf{T}$ for scalar X and Y, then the cdf is defined as

$$P_Z(z) = P_{X,Y}(x,y) = \Pr\{X \leq x, Y \leq y\}$$

and the pdf is found as

$$p_Z(z) = p_{X,Y}(x,y) = \frac{\partial^2}{\partial x \partial y} P_{X,Y}(x,y) = \frac{\partial^2}{\partial y \partial x} P_{X,Y}(x,y).$$

It must again be emphasized that the abstract machinery of Section 2.2.1 is much more general than random variables in $\mathbb{R}^n$. More care is needed if one aims to extend the constructions of cdfs and pdfs to exotic state spaces such as closed manifolds.

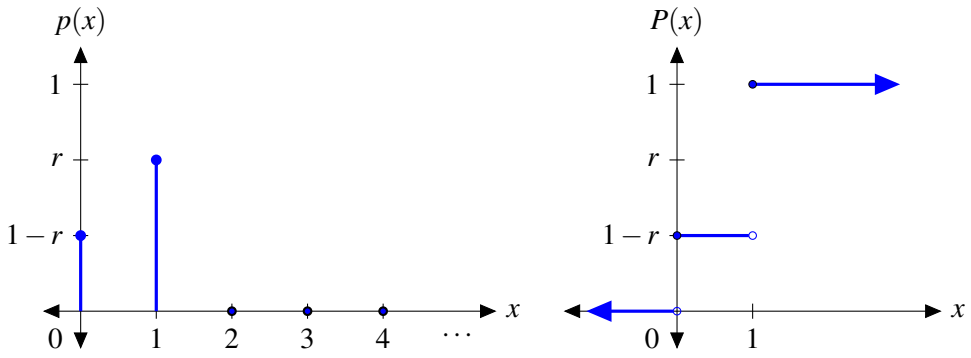
2.2.3 Examples of probability distributions

It is time to look at some examples of random variables that play important roles in sensor fusion. We shall first look at a handful of discrete examples.

■ **Example 2.2 — Bernoulli random variable.** A Bernoulli random variable with parameter r has a binary outcome space: $E = \{0, 1\}$, and its probability distribution is given by

$$p(x) = \begin{cases} 1-r & \text{if } x = 0 \\ r & \text{if } x = 1 \\ 0 & \text{otherwise} \end{cases} \tag{2.4}$$

We write $p(x) = \text{Bernoulli}(x; r)$ to signify this distribution. ■



(a) PMF. (b) CDF.

Figure 2.1: Example of Bernoulli distribution.

■ **Example 2.3 — Binomial random variable.** A binomial random variable with parameters $r \in [0, 1]$ and $n \in \mathbb{N}$ has outcome space $\{0, \ldots, n\}$ and its probability distribution is given by

$$p(x) = \binom{n}{x} r^x (1-r)^{n-x}. \tag{2.5}$$

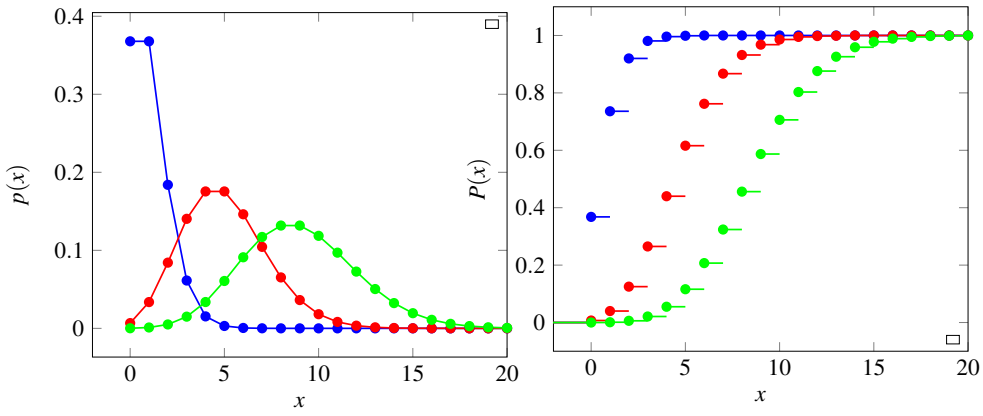
■

■ **Example 2.4 — Poisson random variable.** A Poisson random variable with parameter λ has the countable (that means infinite, but discrete) outcome space $\{0, 1, 2, 3, \ldots\}$ and its probability distribution is given by

$$p(x) = \frac{\lambda^x e^{-\lambda}}{x!} \tag{2.6}$$

We write $p(x) = \text{Poisson}(x; \lambda)$ to signify this distribution. ■

Bernoulli random variables are often used to model whether a target exists or does not exist, or whether a target is detected or not detected. Poisson random variables are often used to model how many false alarms there are in a radar scan. The binomial random variable can be viewed as a generalization of the Bernoulli random variable when there is more than one yes-no question, e.g., two potential targets, or two sensor cells which may or may not contain an observation. According to the Poisson limit theorem, also known as the law of rare events, the binomial distribution can be approximated by the Poisson distribution if n tends towards infinity and r tends towards zero in such a manner that the product nr tends towards λ.



(a) PMF (b) CDF

Figure 2.2: Poisson distribution.

As for continuous random variables, the simplest possible example is the uniform distribution, which for instance often is used in target tracking to model the spatial distributions of false alarms or newborn targets.

■ **Example 2.5 — Uniform random variable.** A uniformly distributed random variable X on the interval $[a,b]$ has the pdf

$$p(x) = \text{Uniform}(x;[a,b]) = \frac{1}{b-a}\chi_{[a,b]}(x) = \begin{cases} \frac{1}{b-a} & \text{if } a \leq x \leq b \\ 0 & \text{otherwise.} \end{cases} \tag{2.7}$$

Its cdf can be written as

$$p(x) = \begin{cases} 0 & \text{if if } x \leq a \\ \frac{x-a}{b-a} & \text{if } a \leq x \leq b \\ 1 & \text{if } x \geq a. \end{cases} \tag{2.8}$$

■

By far the most important class of continuous random variable are Gaussian random variables. In a similar manner as for uniform random variables, the pdf of a Gaussian random variable is given by two parameters. It is possible to parameterize the pdf in different ways.

■ **Example 2.6 — Gaussian random variable.** A Gaussian random variable with expectation μ and variance σ^2 has the pdf

$$p(x) = \mathcal{N}(x;\mu,\sigma^2) = \frac{1}{\sqrt{2\pi}\sigma}\exp\left(\frac{(x-\mu)^2}{2\sigma^2}\right) \tag{2.9}$$

This is known as the moment-based parameterization. Another parameterization is the canonical parameterization. If we define $\eta = \mu/\sigma^2$ and $\lambda = 1/\sigma^2$, then the same pdf can also be expressed as

$$p(x) = \exp\left(\alpha + \eta x - \frac{1}{2}\lambda x^2\right) \quad \text{where} \quad \alpha = -\frac{1}{2}\left(\ln(2\pi) - \ln(\lambda) + \eta^2/\lambda\right). \tag{2.10}$$

The cdf of the Gaussian does not exist in closed form. It is typically expressed in terms of the so-called error function according to

$$P(x) = \frac{1}{2}\left[1 + \text{erf}\left(\frac{x-\mu}{\sigma\sqrt{2}}\right)\right] \quad \text{where} \quad \text{erf}(x) = \frac{1}{\sqrt{\pi}}\int_0^x \exp(-t^2)\text{d}t. \tag{2.11}$$

■

The Gaussian distribution is the most well-known example of a distribution from the so-called exponential family. Another important member of this family is the Gamma distribution.

■ **Example 2.7 — Gamma and χ^2 random variables.** A Gamma random variable with shape parameter k and scale parameter θ has the pdf

$$p(x) = \text{Gamma}(x; k, \theta) = \frac{x^{k-1}\exp(-x/\theta)}{\theta^k \Gamma(k)} \tag{2.12}$$

Again different parameterizations are possible, but the parameterization in terms of shape and scale is sufficient for us. Several other exponential-family distributions are intimately linked to the Gamma distribution. Both the Rayleigh distribution, the exponential distribution and the χ^2-distribution are special cases of the Gamma distribution. The exponential distribution $p(x) = \lambda e^{\lambda t}$ arises as a special case of (2.12) when $k = 1$ and $\theta = 1/\lambda$. The Rayleigh distribution $p(x) = \frac{x}{\sigma^2} e^{-x^2/(2\sigma^2)}$ arises as a special case when $k = 2$ and $\theta = 2\sigma^2$. As for the χ^2 distribution, we say that X is χ^2 distributed with n degrees of freedom if

$$p(x) = \frac{1}{2^{n/2}\Gamma(n/2)} x^{n/2-1} \exp\left(-\frac{x}{2}\right). \tag{2.13}$$

We can see that (2.13) is a special case of (2.12) when the scale parameter is equal to 2 and the shape parameter is equal to $\frac{n}{2}$. For general Gamma and χ^2 random variables the cdf is again non-analytic, while closed-form expressions are easily found for Rayleigh and exponential RVs. Nevertheless, it is often important to find such cdf values, and to solve for x when a cdf value is given, and this can easily be done with Matlab functions such as `chi2cdf` and `chi2inv`, respectively. ■

2.2.4 Sampling from probability distributions

To play with random variables, it is very useful to be able to simulate them. Programming languages such as Matlab comes with support or add-on packages which allow the user to draw samples from well-known probability distributions without putting much thought into how the simulation is done. For examples, to draw a 4×1 vector of independent samples from $\mathcal{N}(x; 0, 1)$ we can use the Matlab command

```
randn(4,1).
```

More complicated random variables can often be simulated by exploiting their relationships to other random variables. For example, we shall later see (Example 2.11 on page 27) that the absolute value of such a Gaussian random variable is a χ^2 distributed random variable with one degree of freedom. Thus, we have a means of simulating such χ^2 distributed random variables.

If we are not so lucky to have such a relationship to exploit, we can still use *cdf inversion*, also known as inverse transform sampling or the Smirnov transform. Consider the problem of generating N independent samples of a random variable X with pdf $p_X(x)$. In this approach, we draw a random N numbers u^i from the uniform distribution $\text{Uniform}(u; [0, 1])$. The we look at the cdf $P_X(x)$. Every sample x^i is then found as the value of x^i such that

$$P_X(x^i) = u^i. \tag{2.14}$$

In other words, we find the samples as $x^i = P_X^{-1}(u^i)$. We see that this approach relies fundamentally on our ability to invert the cdf. This may or may not be possible to do in closed form. However, even if no closed-form solution exist, approximations may be good enough. For example, we can approximate the cdf by "segmented" functions, in the shape of a staircase or a C^0 collection of straight lines.

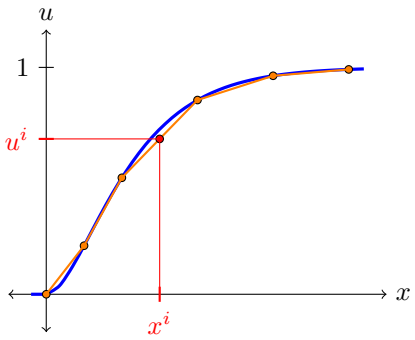


Figure 2.3: Illustration of how the Smirnov transform can be used to sample a random variable even if its inverse cdf has no analytical expression.

2.3 Moments

In many cases, we are not primarily interested in the full pdf of a random variable, but only in certain descriptive measures which give us more tangible information about how the random variable behaves. Among such measures are moments, which are expectations of power of the random variable. The two most well known moments are expectation and variance.

The expectation and variance of a scalar-valued random variable X with realization x are given by the integrals

$$E[X] = \int xp(x)\mathrm{d}x \tag{2.15}$$

$$\mathrm{Var}[X] = E[(X - E[X])^2] = \int (x - E[X])^2 p(x)\mathrm{d}x. \tag{2.16}$$

The notations $\mu = E[X]$ and $\sigma = \sqrt{\mathrm{Var}[X]}$ are commonly used. The generalizations for a vector-valued random variable X with realization $\mathbf{x}$ are given by

$$E[X] = \int \mathbf{x}p(\mathbf{x})\mathrm{d}\mathbf{x} \tag{2.17}$$

$$\mathrm{Var}[X] = E[(X - E[X])(X - E[X])^\mathsf{T}] = \int (\mathbf{x} - E[X])(\mathbf{x} - E[X])^\mathsf{T} p(\mathbf{x})\mathrm{d}\mathbf{x}. \tag{2.18}$$

In the vector case $\mathrm{Var}[X]$ becomes a matrix, and not merely a number. It is known as the covariance matrix and consists of elements of the form

$$\mathrm{Cov}[X_1, X_2] = E[(X_1 - E[X_1])(X_2 - E[X_2])^\mathsf{T}] = \int (x_1 - E[X_1])(x_2 - E[X_2])p(x_1, x_2)\mathrm{d}x_1\mathrm{d}x_2. \tag{2.19}$$

For discrete random variables the integrals are replaced by sums. The expectation and the variance are closely related to the sample mean and the sample variance, which can be calculated for a collection of N samples according to

$$\bar{\mathbf{x}} = \frac{1}{N}\sum_{i=1}^{N} \mathbf{x}_i \tag{2.20}$$

$$\mathbf{P} = \frac{1}{N-1}\sum_{i=1}^{N} (\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x}_i - \bar{\mathbf{x}})^\mathsf{T} \tag{2.21}$$

If the samples are drawn from the distribution $p(x)$, then the mean will converge towards its expectation as the number of samples goes towards infinity. Notice that these sample-dependent quantities in contrast to the expectation and variance are random variables. The distribution of the sample mean can in some cases be calculated analytically, while in other cases it may be intractable. However, as the number of samples goes towards infinity we are guaranteed some regularity thanks to the Central Limit Theorem (CLT).

Theorem 2.3.1 — Central Limit Theorem, Lindeberg-Lévy version. Let $X_1, X_2, \ldots X_n$ be a sequence of i.i.d. random variables with expectation $E[X_i] = \mu$ and $\text{Var}[X_i] = \sigma^2 < \infty$. Let

$$S_n = \frac{1}{n}\sum_{i=1}^{n} X_i.$$

Then, as n approaches infinity, the distribution of the random variable $\sqrt{n}(S_n - \mu)$ converges to $\mathcal{N}(0, \sigma^2)$.

Proof. This is left as an exercise to the reader (Exercise 2.4) since it depends on generating functions, which will be introduced in Section 2.4. ■

The CLT is often used as a justification for assuming that something is Gaussian distributed. In order to assess such justifications it is important to be aware of its requirements. The random variables must be i.i.d. If one wants to analyze the sum of random variables with slightly different distributions, the CLT may break down. Also notice that $\text{Var}[X_i]$ must exist. There exist distributions for which the variance is infinity, and for such distributions the CLT cannot be used.

Since expectation is given by an integral it inherits the linearity property of the integral. Therefore, we can in general write

$$E[\mathbf{A}X + \mathbf{B}Y] = \mathbf{A}E[X] + \mathbf{B}E[Y] \tag{2.22}$$

where X and Y are random vectors and $\mathbf{A}$ and $\mathbf{B}$ are matrices. Variance involves squaring and is therefore not a linear operation. Nevertheless, the squaring operation is so benign that a similar result also can be stated for variance:

$$\text{Var}[\mathbf{A}X + \mathbf{B}Y] = \mathbf{A}\text{Var}(X)\mathbf{A}^\mathsf{T} + \mathbf{B}\text{Var}(Y)\mathbf{B}^\mathsf{T} + \mathbf{A}\text{Cov}(X,Y)\mathbf{B}^\mathsf{T} + \mathbf{B}\text{Cov}(Y,X)\mathbf{A}^\mathsf{T}. \tag{2.23}$$

We notice that variance also is a kind of expectation, which involves taking the expectation of an expectation. It can also be useful to calculate the expectation of an expectation in other contexts. The law of total expectation states that

$$E_X[X] = E_Y[E_X[X|Y]]. \tag{2.24}$$

Here we have introduced subscripts on the expectation operation to specify which random variable it applies to. If the random variables X and Y are independent, then $E_X[X|Y]$ does not depend on Y, and it follows that

$$E_{X,Y}[XY] = E_X[E_Y[XY]] = E_Y[E_X[X|Y]Y] = E_X[X]E_Y[Y]. \tag{2.25}$$

Expectation tends to have a smoothing behavior. For this reason there exist several inequalities that expectations must obey. The most famous such inequality is Jensen's inequality, which states that for an convex function $f(\cdot)$ the following must hold:

$$f(E[X]) \leq E[f(X)] \tag{2.26}$$

Such a convex function could for example be $f(x) = x^2$. Jensen's inequality can be proved by noting that the convexity of $f(\cdot)$ implies that $f(X) \geq f(E[X]) + f'(E[X])(X - E[X])$, which again implies that $f(X) \geq f(E[X])$ because $E[X - E[X])] = 0$.

We can also study moments of higher order than expectation (first-order) and variance (second-order). In particular, the fourth order moment is often used to measure how heavytailed a distribution is. By normalizing this according to the variance squared, we get the kurtosis

$$\text{Kurt}[X] = \frac{E[(X-\mu)^4]}{\sigma^4} \tag{2.27}$$

which is the standard measure for heavytailedness. For the univariate Gaussian distribution it can be shown that the kurtosis is 3. Distributions with higher kurtosis are known as leptokurtic or heavytailed, while distributions with lower kurtosis are known as platykurtic. We can also analyse how much a distribution differs from the shape of the Gaussian by evaluating its skewness, which is defined as

$$\gamma = \frac{E[(X-\mu)^3]}{\sigma^3}. \tag{2.28}$$

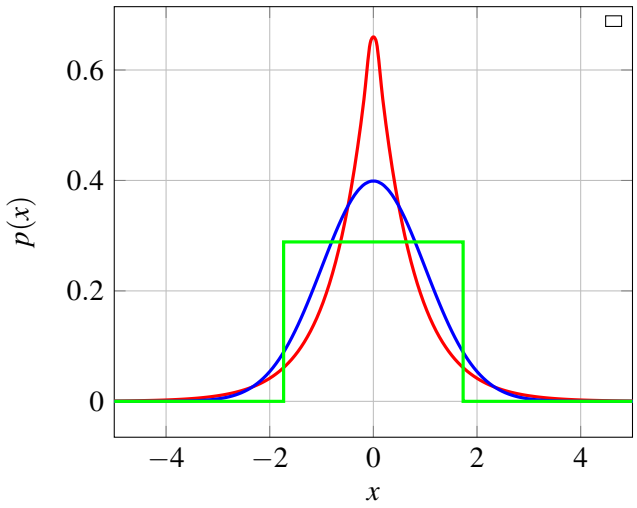


Figure 2.4: Kurtosis values for three symmetric distributions that have mean of zero and variance of one: Laplace distribution (red), normal distribution (blue) and uniform distribution (green).

A variety of information measures which describe random variables can be expressed and understood as expectations. These include the entropy

$$H[X] = E[-\ln(p(X))] \tag{2.29}$$

and the Kullback-Leibler divergence

$$D_{\text{KL}}(p,q) = E\left[-\ln\left(\frac{q(X)}{p(X)}\right)\right] \tag{2.30}$$

which provides a measure for how similar an approximating distribution $q(X)$ is to the true distribution $p(X)$. Such measures play a central role in many modern estimation methods.

2.4 Generating functions and transformations of RVs

We recall from basic control theory that sometimes it is easier to manipulate functions through their Laplace and Fourier transforms, than to work with the functions themselves. This is for example often the case when solving differential equations. In the same way, several transform-domain representations of probability distributions exist, and do often provide convenient representations. These are known as generating functions.

A generating function of a random variable is the expectation of a certain transformation of that random variable. The three most commonly encountered generating functions are known as the characteristic function, the moment-generating function and the probability-generating function. The first two are used for continuous random variables, and are in the scalar case given by

$$\Phi_X(\omega) = E_X[e^{\mathrm{i}\omega x}] = \int_{-\infty}^{\infty} p(x)e^{\mathrm{i}\omega x}\mathrm{d}x \tag{2.31}$$

$$M_X(s) = E_X[e^{sx}] = \int_{-\infty}^{\infty} p(x)e^{sx}\mathrm{d}x. \tag{2.32}$$

The latter function is used for discrete random variables, and is (again in the scalar case) given by

$$G(t) = E_X[t^X] = \sum_{n=-\infty}^{\infty} p(x_n)t^{x_n}. \tag{2.33}$$

All of these representations share three important properties:

1. The generating function determines the distribution and vice versa.
2. The generating function of a sum of independent random variables is the product of the generating functions.
3. The moments can be found by differentiating the generating function.

Property 1 should not come as a surprise. We see that $\Phi_X(\omega)$ and $M_X(s)$ work in a manner very similar to Fourier and Laplace transforms, respectively, and we know that the same principle applies to Fourier and Laplace transforms. We recognize the probability generating function as the z-transform of the pmf.

Property 2 is the most important reason why one should be familiar with generating functions: Whenever a random variable is a sum of two or more other random variables with known distributions, generating functions is a convenient tool to find the distribution of the random variable in question. Property 3 tends to be the rationale under which generating functions are presented in basic statistics courses. It is also important: It can be much more convenient to find moments by derivation than through the alternative route of integration.

■ **Example 2.8 — Sum of Bernoulli and Poisson.** Let $X \sim \text{Bernoulli}(x;p)$ and let $Y \sim \text{Poisson}(y;\lambda)$, and let $S = X + Y$. What is the probability distribution of S?
Solution: The probability generating functions of X and Y can be found as

$$G_X(t) = 1 - r + rt \tag{2.34}$$

$$G_Y(t) = \exp(\lambda(t-1)). \tag{2.35}$$

The probability generating function of S is therefore

$$G_S(t) = (1-r+rt)\exp(\lambda(t-1)) = (1-r)\exp(\lambda(t-1)) + rt\exp(\lambda(t-1)) \tag{2.36}$$

We get the probabilities of S by differentiating the probability generating function. Let us first notice that

$$\frac{\mathrm{d}^k}{\mathrm{d}t^k}\left[te^{\lambda t}\right] = \frac{\mathrm{d}^{k-1}}{\mathrm{d}t^{k-1}}\left[e^{\lambda t} + \lambda te^{\lambda t}\right] = \ldots = k\lambda^{k-1}e^{\lambda t} + \lambda^k te^{\lambda t}. \tag{2.37}$$

Based on this we find the probability that $S = k$ according to

$$\begin{aligned} p(k) &= \frac{1}{k!}\frac{\mathrm{d}^k}{\mathrm{d}t^k}G_S(t)\Big|_{t=0} \\ &= \frac{1}{k!}\left((1-r)e^{-\lambda}\frac{\mathrm{d}^k}{\mathrm{d}t^k}\left[e^{\lambda t}\right] + re^{-\lambda}\frac{\mathrm{d}^k}{\mathrm{d}t^k}\left[te^{\lambda t}\right]\right)\Big|_{t=0} \\ &= \frac{1}{k!}\left(\lambda^k(1-r)e^{-\lambda} + rk\lambda^{k-1}e^{-\lambda}\right). \end{aligned} \tag{2.38}$$

■

■ **Example 2.9 — Sum of Exponentials.** Let X_i, $i = 1,\ldots,N$ be N i.i.d. exponential RVs with parameter λ, and define $Y = \sum_{i=1}^{N} X_i$. What is the probability distribution of Y?
Solution: The moment-generating function for each of the i.i.d. exponentials is

$$M_X(s) = \lambda\int_0^\infty e^{(s-\lambda)x}\mathrm{d}x = \frac{\lambda}{\lambda - s} \quad \text{if} \quad s < \lambda. \tag{2.39}$$

According to Property 2 of the generating function this implies that

$$M_Y(s) = \left(\frac{\lambda}{\lambda - s}\right)^N. \tag{2.40}$$

If we were to calculate the inverse Laplace transformation of such an expression, we would end up with terms looking something like $x^N\exp(-\lambda x)$. Since this is precisely what we have in the Gamma function, let us check its moment-generating function:

$$M_Y(s) = \frac{1}{\theta^k\Gamma(k)}\int_0^\infty x^{k-1}\exp\left(-x\left(\frac{1}{\theta} - s\right)\right)\mathrm{d}x = \frac{1}{\theta^k\Gamma(k)}\Gamma(k)\left(\frac{1}{\frac{1}{\theta} - s}\right)^k = \left(\frac{1}{1/-\theta s}\right)^k. \tag{2.41}$$

By comparing (2.40) with (2.41) we see that the sum of N exponentials becomes a Gamma distributed random variable with scale parameter $1/\lambda$ and shape parameter N. ■

While all the three generating functions are valid for a continuous random variable, only the probability-generating function is used for discrete RVs. Generalization to vector-valued random variables is straightforward. For a vector-valued random variable X the characteristic function is

$$\Phi_X(\boldsymbol{\omega}) = E_X[e^{\mathrm{i}\boldsymbol{\omega}^\mathsf{T}\mathbf{x}}] = \int_\infty p(\mathbf{x})e^{\mathrm{i}\boldsymbol{\omega}^\mathsf{T}\mathbf{x}}\mathrm{d}\mathbf{x}. \tag{2.42}$$

While we now have a tool for finding the distributions of sums of random variables, we lack a systematic methodology for discovering the distributions that result when a scalar or vector-valued random variable is subject to a non-linear transformation. While closed-form solutions in most such cases are unattainable, fundamental formulas nevertheless exist.

Theorem 2.4.1 — Nonlinear transformations of random variables. Suppose that $\mathbf{y} = \mathbf{f}(\mathbf{x})$ where $\mathbf{h} : \mathbb{R}^n \to \mathbb{R}^n$ and $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$. Denote the pdf's of $\mathbf{x}$ and $\mathbf{y}$ by $g(\mathbf{x})$ and $h(\mathbf{y})$, respectively. Then we have that

$$h(\mathbf{y}) = \sum_i g(\mathbf{f}_i^{-1}(\mathbf{y}))\,|\det(\mathbf{F}_i^{-1}(\mathbf{y}))|$$

where $\mathbf{f}_i^{-1}(\mathbf{y})$ range over all solutions of $\mathbf{y} = \mathbf{f}(\mathbf{x})$ with respect to $\mathbf{x}$, and $\mathbf{F}_i^{-1}(\mathbf{y})$ is the corresponding Jacobian matrix of the inverse mapping $\mathbf{f}_i^{-1}(\mathbf{y})$.

Proof. For a proof in the 2-dimensional case, see [70] pages 199-201. ■

The sum in Theorem 2.4.1 reduces to a single term whenever $\mathbf{f}$ is invertible.

■ **Example 2.10 — Amplitude and phase of circularly symmetric Gaussian.** Let $\mathbf{x} = [x_1, x_2]^\mathsf{T}$ be a 2-dimensional random variable given by

$$g(\mathbf{x}) = \mathcal{N}(\mathbf{x}_1\,; 0, \sigma^2)\mathcal{N}(\mathbf{x}_2\,; 0, \sigma^2) = \mathcal{N}(\mathbf{x}; \mathbf{0}, \sigma^2\mathbf{I}). \tag{2.43}$$

In the last expression we have introduced the notation of the multivariate Gaussian, which will be extensively studied in Chapter 3. We want to find the pdf of the 2-dimensional random variable $\mathbf{y}$ given by $\mathbf{y} = \mathbf{f}(\mathbf{x})$ where $\mathbf{f} : \mathbb{R}^2 \to [0,\infty) \times [0, 2\pi)$ is given by

$$\mathbf{f}(\mathbf{x}) = \begin{bmatrix} \|\mathbf{x}\|_2 \\ \text{atan2}(x_2, x_1) \end{bmatrix}. \tag{2.44}$$

Solution: First we notice that $\mathbf{f}$ is invertible on the given domain, since it simply is a conversion from Cartesian to polar coordinates. We can therefore drop the subscript i. Let us denote the components of $\mathbf{y}$ by r and θ, so that $\mathbf{y} = [r, \theta]^\mathsf{T}$. The inverse mapping and its Jacobian are given by

$$\mathbf{f}^{-1}(\mathbf{y}) = \begin{bmatrix} r\cos\theta \\ r\sin\theta \end{bmatrix} \quad \text{and} \quad \mathbf{F}^{-1}(\mathbf{y}) = \begin{bmatrix} \cos\theta & -r\sin\theta \\ \sin\theta & r\cos\theta \end{bmatrix}. \tag{2.45}$$

It is easy to see that $|\mathbf{F}^{-1}(\mathbf{y})| = r$, and it follows that

$$h(\mathbf{y}) = \frac{r}{2\pi\sigma^2}\exp\left(-\frac{1}{2\sigma^2}\left(r^2\cos^2\theta + r^2\sin^2\theta\right)\right) = \frac{r}{2\pi\sigma^2}\exp\left(-\frac{r^2}{2\sigma^2}\right) \tag{2.46}$$

While the expression in (2.46) technically solves the given problem, we are not entirely satisfied before we have the marginal pdfs of r and θ. Since only r is present in (2.46), and since θ is defined on an interval of length 2π, we do not need to perform any aditional calculations to find these: We simply factorize the joint density as $h(\mathbf{y}) = p_r(r)p_\theta(\theta)$ where

$$p(r) = \frac{r}{\sigma^2}\exp\left(-\frac{r^2}{2\sigma^2}\right) = \text{Rayleigh}(r\,; \sigma^2) \tag{2.47}$$

$$p(\theta) = \frac{1}{2\pi}\chi_{[0,2\pi]}(\theta) = \text{Uniform}(\theta\,; [0, 2\pi]). \tag{2.48}$$

■

■ **Example 2.11 — Square of zero-mean univariate Gaussian.** If $X \sim \mathcal{N}(0,1)$, what is then the pdf of $Y = X^2$?

Solution: In this case, the mapping $\mathbf{f}$ in Theorem 2.4.1 is not invertible. For any realization of Y, we have two possible realizations of X:

$$\mathbf{f}_1^{-1} = \sqrt{y} \quad \text{and} \quad \mathbf{f}_2^{-1} = -\sqrt{y}. \tag{2.49}$$

The corresponding Jacobians are

$$\mathbf{F}_1^{-1}(y) = \frac{1}{2\sqrt{y}} \quad \text{and} \quad \mathbf{F}_2^{-1}(y) = \frac{-1}{2\sqrt{y}}. \tag{2.50}$$

By stitching this together, we arrive at

$$h(y) = \frac{1}{\sqrt{2\pi}}\left[\frac{1}{2\sqrt{y}}e^{-\frac{1}{2}(\sqrt{y})^2} + \frac{1}{2\sqrt{y}}e^{-\frac{1}{2}(-\sqrt{y})^2}\right] = \frac{1}{\sqrt{2\pi y}}e^{-y/2}. \tag{2.51}$$

We recognize this as a χ^2 distribution with 1 degree of freedom. ■

2.5 Frequentist and Bayesian approaches to probability

While everyone can agree on the fundamental framework outlined in the previous section, significant controversy exists when it comes to the more philosophical interpretation of what probability really is.

First, there is the question of whether randomness actually occur in nature, or whether randomness only serves as a model for stuff we do not have any better model for. In target tracking, the latter interpretation is much more relevant than the former, but this entails that we always must consider whether randomness actually provides a realistic model.

Second, there is a distinction between frequentist and Bayesian interpretation of probability. According to a die-hard frequentist, all probabilities should have an interpretation of a frequency of some sort. For example, if it rains in 250 out of 365 days in Bergen, then the probability of rain in Bergen is about 0.68. In contrast, most Bayesians are willing to consider statistical models that involve subjective assignments of probability. Such information is encoded in a prior distribution $p(X)$ which together with the likelihood $p(z|x)$ yields the posterior distribution

$$p(x|z) \propto p(z|x)p(x) \tag{2.52}$$

This allows the Bayesian to say that the unknown x has so and so probability for being this or that, or within a given interval. Frequentists will do no such thing. Instead a frequentist will analyze how plausible the data are given x. The philosophical difference between the two schools can be seen in the different interpretations of frequentist *confidence interval* and Bayesian *credible intervals*. A 95% confidence interval is constructed so that if the experiment is repeated several times, then it will cover the true and deterministic value of x 95% of the time. A 95% credible interval will contain the true and random value of x 95% of the time.

The capability of treating estimation in purely probabilistic terms is a huge advantage for the Bayesian approach. The Bayesian can always present the full posterior as a solution to his estimation problem, and ask the customer what information she would like to see extracted from it. The caveat is of course that the prior distribution, which necessarily has a subjective element, normally will play an important role in shaping the posterior. For some problems it may be appropriate to choose a so-called noninformative prior, which is designed to minimize the amount of information imposed by the prior. In other cases, physical models do indeed provide prior knowledge that it would be rather silly not to utilize. This is typically the case in sensor fusion.

2.5.1 Bayes and conditional probability for continuous RVs

While Bayes' rule itself is not a controversial result, it is nevertheless not obvious for continuous random variables, but something that must be proved.

Theorem 2.5.1 — Bayes' rule for pdfs. Let the continuous random variables X and Z have the joint distribution $p(x,z)$. Then

$$p(x|z) = \frac{p(z|x)p(x)}{p(z)} \tag{2.53}$$

Proof. For simplicity assume that both X and Z are scalar. Let us first define the events

$$\begin{aligned} A &= \{\, z \le Z \le z + \Delta z \,\}. \\ B &= \{\, x \le X \le x + \Delta x \,\}. \end{aligned} \tag{2.54}$$

Let us denote all the cdfs of x and z by $P(\cdot)$, with and without conditioning on A or B. Bayes' law

for probabilities yields

$$\frac{\Pr\{A\,|\,B\}}{\Pr\{A\}} = \frac{\Pr\{B\,|\,A\}}{\Pr\{B\}} \tag{2.55}$$

We can rewrite this expression in terms of cdfs as follows.

$$\frac{\dfrac{P(z+\Delta z\,|\,B)-P(z\,|\,B)}{\Delta z}}{\dfrac{P(z+\Delta z)-P(z)}{\Delta z}} = \frac{\dfrac{P(x+\Delta x\,|\,A)-P(x\,|\,A)}{\Delta x}}{\dfrac{P(x+\Delta x)-P(x)}{\Delta x}} \tag{2.56}$$

We have included division by Δz and Δx above and below the original fraction bar because we obviously intend to convert the cdfs to pdfs, which will involve differentiation. Let us then define the conditional pdfs according to

$$\begin{aligned} p(z\,|\,x) &= \lim_{\Delta x\to 0} p(z\,|\,B) = \lim_{\Delta z\to 0}\lim_{\Delta x\to 0} \frac{P(z+\Delta z\,|\,B)-P(z\,|\,B)}{\Delta z} \\ p(x\,|\,z) &= \lim_{\Delta z\to 0} p(x\,|\,A) = \lim_{\Delta x\to 0}\lim_{\Delta z\to 0} \frac{P(x+\Delta x\,|\,A)-P(x\,|\,A)}{\Delta x}. \end{aligned} \tag{2.57}$$

If we now take the double limit of (2.56) for $\Delta x \to 0$ and $\Delta z \to 0$ we can insert these pdfs in the numerators, while the denominators obviously turn into $p(z)$ and $p(x)$. Thus, (2.56) becomes

$$\frac{p(z\,|\,x)}{p(z)} = \frac{p(x\,|\,z)}{p(x)} \tag{2.58}$$

which is nothing more than a restatement of Bayes' rule for pdfs. ■

It is important to keep in mind that all results concerning pdf's of continuous random variables, including Bayes and conditional probability, are based on differentiation of proper probabilities. Differentiation is always relative to the metric properties, i.e., parametrization and units, of the underlying space, which may be $\mathbb{R}$, $\mathbb{R}^d$ or something more exotic, e.g., a sphere. The choice of coordinate system may not only affect the values of the pdf's, but also their overall shape.

2.6 Estimators

Estimation is the task of inferring knowledge about an unknown quantity x from data z which are related to x. In the probabilistic paradigm, the relationship between z and x is in the form of a probabilistic model $p(z\,|\,x)$. In the frequentist approach, this is all that is given. The Bayesian approach, prior knowledge about x is also given in the form of another probabilistic model $p(x)$, and one must then weigh the two models against each other to find an estimate of x.

For a given estimation problem, an *estimator* is a procedure that attempts to guess a concrete value for what x is based on the data. The output of the estimator is called an *estimate*.

Definition 2.6.1 — Estimator. Let $\mathscr{Z}$ and $\mathscr{X}$ be the spaces in which z and x belong, respectively. An estimator is a function $\theta : \mathscr{Z} \to \mathscr{X}$ so that $\theta(z)$ gives an estimate of x. We use a hat to denote the estimate, i.e., $\hat{x} = \theta(z)$.

Estimators can be categorized according to various categories. First, there is of course the divide between frequentist and Bayesian estimators. We may also distinguish between *maximizing* and *averaging* estimators. A maximizing estimator aims to find the one best value of x given z. An averaging estimator aims to find a value of x which is representative of our knowledge about x given z. While maximizing estimators can be defined in both frequentist and Bayesian estimation, the concept of an averaging estimator necessarily involves elements of Bayesian thinking. If one wants to have such tools available, it is hard remain a die-hard frequentist.[1]

[1] "Everyone is either a Bayesian or a closet Bayesian", Josef Knoll.

2.6.1 Maximizing estimators

Two well-known maximizing estimators are the maximum likelihood (ML) and maximum *a posteriori* (MAP) estimators. The ML estimator is given by

$$\hat{x} = \arg\max_x p_{Z|X}(z|x). \tag{2.59}$$

while the MAP estimator is given by

$$\hat{x} = \arg\max_x p_{X|Z}(x|z) = \arg\max_x p_{Z|X}(z|x)p_X(x). \tag{2.60}$$

In some special cases closed-form expressions for the ML or MAP estimators exist. It does probably not come as a big surprise that such cases include estimation of expectation and covariance for a Gaussian distribution. Another example is given in Example 2.12. In general, however, numerical search techniques are needed to implement these estimators. The menu that such techniques can be chosen from is fairly daunting, and a solid understanding of the objective functions in (2.59) or (2.60) is essential, both to succeed at all, and to achieve acceptable computational efficiency.

To which extent are the ML and MAP estimators reliable? The answer is to a larger extent affirmative for the ML estimator than for the MAP estimator. The ML estimator has some nice properties that statisticians refer to as *consistency* and *efficiency*. The former means that given enough data it will always recover the correct parameter. The latter means that it as it approaches that limit will attain a higher accuracy than any other kind of estimator that utilizes the same information.

For the MAP estimator we need to be more careful, since it also involves a prior distribution and Bayes' rule. Recalling that pdf's are only defined by differentiation relative to the coordinate system, it is clear that a change of coordinates can alter the location of the peak of a pdf. Since the MAP estimator maximizes the posterior pdf, it is therefore clear that the MAP estimator may not be invariant under coordinate transformations. This should not, however, be taken as a criticism of the Bayesian paradigm. A bona-fide Bayesian will never think of the MAP estimator as the solution to her estimation problem. In the Bayesian mindset, the entire posterior distribution is the solution, and more or less reliable information about this solution can be extracted by various estimators. For discrete estimation problems, also known as classification problems, the picture is somewhat different. Then there is a (hopefully nonzero) probability that the estimator hits the right value or class, and it can be shown that the MAP estimator has the highest success rate of all possible estimators, see Exercise 2.7.

2.6.2 Estimators as random variables

Since an estimator depends on the data, and the data both in Bayesian and frequentist schools are random variables, estimators are random variables as well. This means that a given estimator for a given model has a particular distribution. It is important to be aware of this for several reasons. Often, one wants to know the mean square error (MSE) of an estimator, and the MSE is given by this distribution. Furthermore, knowledge about an estimator's distribution can be important in subsequent processing of the estimate. For example, if two different estimates are to be combined in a sensor fusion system, then we would typically put most weight on the estimate with the lowest MSE.

■ **Example 2.12 — ML estimator of Rayleigh distribution.** Let $\mathbf{z} = [z_1, \ldots, z_M]^\mathsf{T}$ consist of i.i.d. samples from a Rayleigh distribution:

$$p(\mathbf{z}\,|\,\eta) = \prod_{i=1}^{M} \frac{z_i}{\eta} \exp\left(-\frac{z_i^2}{\eta^2}\right). \tag{2.61}$$

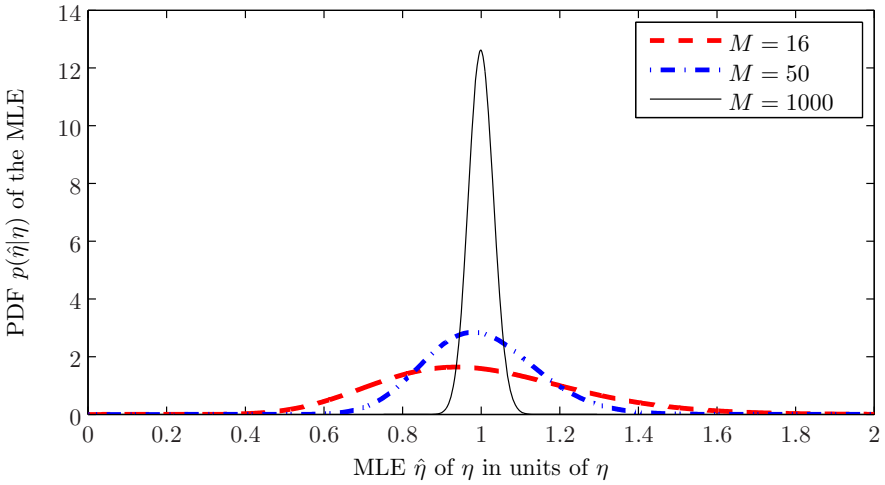


Figure 2.5: Distribution of the Rayleigh MLE for different sample sizes. Notice that a rather large number of samples is needed if η is to be estimated with accuracy of say 5%.

By differentiating the logarithm of $p(\mathbf{z}\,|\,\eta)$ and equating the derivative to zero, we get the ML estimator

$$\hat{\eta} = \frac{1}{2M}\sum_{i=1}^{M} z_i^2. \tag{2.62}$$

It can be shown that this quantity has a Gamma distribution. In Exercise 2.3 you are asked to show that each z_j^2 has an exponential distribution. Furthermore, we know from Example 2.9 that such a sum of exponential random variables is Gamma distributed. Through these steps we arrive at

$$p(\hat{\eta}\,|\,\eta) = \text{Gamma}\left(\hat{\eta}\,; M, \frac{2\eta}{M}\right) = \frac{M^M}{\Gamma(M)}\left(\frac{\hat{\eta}}{\eta}\right)^M \exp\left(-M\frac{\hat{\eta}}{\eta}\right). \tag{2.63}$$

This result is of central importance in detection theory, because we can use it to tune detection probabilities and false alarm rates of radar detectors. ∎

Related to the concept of estimators is the concept of *statistics*. A statistic is a single measure of some attribute of a sample. Estimators are statistics, but we can also construct statistics that not necessarily correspond to a meaningful estimator. As an example, the term $\sum_{i=1}^{M} z_i^2$ in Example 2.12 is a statistic. When we talk about statistics in this sense, we are primarily interested in *sufficient statistics*.

Definition 2.6.2 — Sufficient statistic. A sufficient statistic $g(\mathbf{z})$ for the parameter $\mathbf{x}$ summarizes the information about $\mathbf{x}$ contained in the data $\mathbf{z}$.

Returning to Example 2.12, we see that the term $\sum_{i=1}^{M} z_i^2$ is also a sufficient statistic for the parameter η. This is evident, because in (2.62) the individual data values z_j only occur in this expression. In other words, there is no dependence of the data which are not encapsulated by this term, and the term is then by definition a sufficient statistic. More generally, according to the Neyman-Fisher factorization theorem, it holds that $g(\mathbf{z})$ is a sufficient statistic if the likelihood can be factorized as $f(\mathbf{z}\,|\,\mathbf{x}) = f_1(g(\mathbf{z}),\mathbf{x})f_2(\mathbf{z})$. By a slight abuse of terminology, the parameters required to describe a given distribution, e.g., expectation and covariance of a Gaussian, are sometimes referred to as sufficient statistic. This is abusive because these are not functions of the data. On the other hand, the sample mean and the sample covariance *are* statistics.

2.6.3 LS and MMSE estimators

The least squares (LS) estimator and the minimum mean square error (MMSE) estimator both represent philosophies different from the the ML or MAP estimators. Consider an estimation problem of the form $\mathbf{z} = \mathbf{h}(\mathbf{x}) + \mathbf{w}$ where $\mathbf{w}$ is an unknown vector of measurement noise. The least squares estimator is

$$\hat{\mathbf{x}}_{\text{LS}} = \arg\min_{\mathbf{x}} \|\mathbf{z} - \mathbf{h}(\mathbf{x})\|_2^2. \tag{2.64}$$

It is evident from (2.64) that the LS estimator does not make any assumptions about the measurement "noise" $\mathbf{w}$. It is therefore of a non-probabilistic nature, and is a prime candidate to be considered in problems where a probabilistic model is not available. If $\mathbf{w}$ consists of i.i.d. Gaussian samples, then the LS estimator becomes identical to the MLE.

The idea of minimizing quadratic cost functions also plays a central role in the Bayesian paradigm. In accordance with the Bayesian philosophy of accounting for all uncertainty, the central concept in Bayesian decision theory is to make decisions that minimize given risk measures. An estimator that can be shown to minimize some risk function, is said to be Bayes-optimal with regard to that risk function. This is a very general framework. The decisions can in principle be any actions that are based on observing the data. The risk function can be any quantification of the consequences of the decision. Within the limited scope of estimation, however, the decisions will typically be nothing more than choosing a particular possibility of the estimatee as the estimate. In mathematical terms, if $l(\hat{\mathbf{x}}, \mathbf{x})$ is a loss function describing the deviation between the estimatee θ and our guess of its value $\hat{\theta}$, then the corresponding Bayes-optimal estimator is

$$\hat{\mathbf{x}}_l = \arg\min_{\hat{\mathbf{x}}} E_{\mathbf{x}}[l(\hat{\mathbf{x}}, \mathbf{x})]. \tag{2.65}$$

The most popular risk function is the expectation of the squared error, which in the vector case means the expectation of the function

$$l(\hat{\mathbf{x}}, \mathbf{x}) = (\hat{\mathbf{x}} - \mathbf{x})^{\mathsf{T}}(\hat{\mathbf{x}} - \mathbf{x}). \tag{2.66}$$

The estimator that results is known as the minimum mean square error (MMSE) estimator, which also is known as the expected *a posteriori* estimator, for reasons that will become obvious in the following theorem.

Theorem 2.6.1 — MMSE estimator. The MMSE estimator is given by

$$\hat{\mathbf{x}}_{\text{MMSE}} = \arg\min_{\hat{\mathbf{x}}} E\left[(\hat{\mathbf{x}} - \mathbf{x})^{\mathsf{T}}(\hat{\mathbf{x}} - \mathbf{x})\right] = E[\mathbf{x}|\mathbf{z}] = \int \mathbf{x} p(\mathbf{x}|\mathbf{z}) \mathrm{d}\mathbf{x}. \tag{2.67}$$

Proof. If $\hat{\mathbf{x}}_{\text{MMSE}}$ minimizes $E[l(\hat{\mathbf{x}}, \mathbf{x}) \,|\, \mathbf{z}]$ for all $\mathbf{z}$, then it will also minimize $E[l(\hat{\mathbf{x}}, \mathbf{x})]$. We proceed to show that it indeed does that by means of differentiation.

$$\frac{\partial E[(\hat{\mathbf{x}} - \mathbf{x})^{\mathsf{T}}(\hat{\mathbf{x}} - \mathbf{x}) \,|\, \mathbf{z}]}{\partial \hat{\mathbf{x}}} = E\left[\frac{\partial (\hat{\mathbf{x}} - \mathbf{x})^{\mathsf{T}}(\hat{\mathbf{x}} - \mathbf{x})}{\partial \hat{\mathbf{x}}} \,|\, \mathbf{z}\right] = E\left[-2(\hat{\mathbf{x}} - \mathbf{x})^{\mathsf{T}} \,|\, \mathbf{z}\right] \tag{2.68}$$

To identify the minimizer of the MSE we equate this with zero, which yields

$$\mathbf{0} = E\left[\hat{\mathbf{x}} - \mathbf{x} \,|\, \mathbf{z}\right] = E\left[\hat{\mathbf{x}} \,|\, \mathbf{z}\right] - E\left[\mathbf{x} \,|\, \mathbf{z}\right]. \tag{2.69}$$

Since $\hat{\mathbf{x}}$ is a function of the data we have that $\hat{\mathbf{x}} = E\left[\hat{\mathbf{x}} \,|\, \mathbf{z}\right]$ and it follows that $\hat{\mathbf{x}} = E\left[\mathbf{x} \,|\, \mathbf{z}\right]$. ■

2.6.4 Bias, MSE and variance of estimators

We generally want estimators that have low bias and MSE. Let $\tilde{\mathbf{x}} = \mathbf{x} - \hat{\mathbf{x}}$ be the estimation error. The bias of an estimator is then defined as the expectation $E[\tilde{\mathbf{x}}]$. We say that an estimator is *unbiased* if $E[\tilde{\mathbf{x}}] = 0$. The MMSE estimator is always unbiased. The ML and MAP estimators are, on the other hand, in general biased. Unbiasedness is obviously desirable in general. However, there may exist biased estimators with lower MSE than the best unbiased estimator, and there may exist estimation problems where the requirement of unbiasedness will lead to unacceptable degradation of the MSE.

In the scalar case, the variance and MSE of an estimator are given by

$$\mathrm{Var}(\hat{x}) = E[(\hat{x} - E[\hat{x}])^2] \ , \ \mathrm{MSE}(\hat{x}) = E[(\hat{x} - x)^2]. \tag{2.70}$$

From this it follows that the MSE can be decomposed into variance and bias as follows:

$$\mathrm{MSE}(\hat{x}) = \mathrm{Var}(\hat{x}) + \mathrm{Bias}(\hat{x}, x)^2. \tag{2.71}$$

In the vector case, the variance becomes a matrix:

$$\mathrm{Cov}(\hat{\mathbf{x}}) = E\left[(\hat{\mathbf{x}} - E[\hat{\mathbf{x}}])(\hat{\mathbf{x}} - E[\hat{\mathbf{x}}])^\mathsf{T}\right], \tag{2.72}$$

The MSE is on the other hand a scalar number also in the vector case, as already indicated by (2.67):

$$\mathrm{MSE}(\hat{\mathbf{x}}) = E\left[\|\mathbf{x} - \hat{\mathbf{x}}\|_2^2\right] = E\left[(\hat{\mathbf{x}} - \mathbf{x})^\mathsf{T}(\hat{\mathbf{x}} - \mathbf{x})\right] = \mathrm{tr}\left(E[(\hat{\mathbf{x}} - \mathbf{x})(\hat{\mathbf{x}} - \mathbf{x})^\mathsf{T}]\right). \tag{2.73}$$

2.6.5 LMMSE estimators

When the MMSE estimator is too complicated we may settle for the best linear estimator.

Theorem 2.6.2 — LMMSE estimator. The best estimator of the form $\hat{\mathbf{x}} = \mathbf{A}\mathbf{z} + \mathbf{b}$ that minimizes $\mathrm{MSE}(\hat{\mathbf{x}}) = E\left[\|\mathbf{x} - \hat{\mathbf{x}}\|_2^2\right]$ is given by

$$\hat{\mathbf{x}} = E[\mathbf{x}] + \mathrm{Cov}(\mathbf{x}, \mathbf{z})\mathrm{Cov}(\mathbf{z})^{-1}(\mathbf{z} - E[\mathbf{z}]). \tag{2.74}$$

Proof. We prove the case where the estimatee is a scalar x, i.e., when the estimator is of the form $\hat{x} = \mathbf{a}^\mathsf{T}\mathbf{z} + b$. First we find what b should be. The estimation error and its square can be written

$$\tilde{x} = x - \mathbf{a}^\mathsf{T}\mathbf{z} - b \tag{2.75}$$

$$\tilde{x}^2 = (x - \mathbf{a}^\mathsf{T}\mathbf{z})^2 - 2b(x - \mathbf{a}^\mathsf{T}\mathbf{z}) + b^2, \tag{2.76}$$

respectively. The MSE is

$$E[\tilde{x}^2] = E[(x - \mathbf{a}^\mathsf{T}\mathbf{z})^2] - 2b(E[x] - \mathbf{a}^\mathsf{T}E[\mathbf{z}]) + b^2. \tag{2.77}$$

To find the value of b that minimizes this we calculate the derivative with respect to b and equate it with zero. This results in

$$b = E[x] - \mathbf{a}^\mathsf{T}E[\mathbf{z}]. \tag{2.78}$$

We could also have arrived at this result by requiring that LMMSE estimator should be unbiased. It is easy to see that if we insert (2.78) into the expression for $\tilde{x}$ in (2.75) then the bias becomes zero.

We proceed to look for the optimal vector $\mathbf{a}$. By demanding the derivative of the MSE with respect to $\mathbf{a}$ to be zero, we obtain

$$\frac{\partial}{\partial \mathbf{a}} E[\tilde{x}^2] = \frac{\partial}{\partial \mathbf{a}} E\left[(x - E[x] - \mathbf{a}(\mathbf{z} - E[\mathbf{z}]))^2\right] = 2E[\tilde{x}(\mathbf{z} - E[\mathbf{z}])^\mathsf{T}] = 0. \tag{2.79}$$

The second equality follows from the chain rule. The result that $E[\tilde{x}(\mathbf{z} - E[\mathbf{z}])^\mathsf{T}] = 0$ is known as the orthogonality principle. By further analyzing what this entails we obtain

$$\begin{aligned} E[\tilde{x}\mathbf{z}^\mathsf{T}] &= E\left[\left(x - E[x] - \mathbf{a}^\mathsf{T}(\mathbf{z} - E[\mathbf{z}]\right)(\mathbf{z} - E[\mathbf{z}])^\mathsf{T}\right] \\ &= \mathrm{Cov}[x, \mathbf{z}] - \mathbf{a}^\mathsf{T}\mathrm{Cov}[\mathbf{z}] = 0. \end{aligned} \tag{2.80}$$

This leads to the optimal choice of

$$\mathbf{a} = \mathrm{Cov}[x, \mathbf{z}]\,\mathrm{Cov}[\mathbf{z}]^{-1}. \tag{2.81}$$

Combining this with the expression for b yields the formula in the theorem. ■

It can also be seen that the matrix MSE of the LMMSE estimator is given by

$$\begin{aligned} E[\tilde{\mathbf{x}}\tilde{\mathbf{x}}^\mathsf{T}] =& E\Big[\left(\mathbf{x} - E[\mathbf{x}] - \mathrm{Cov}[\mathbf{x}, \mathbf{z}]\mathrm{Cov}[\mathbf{z}]^{-1}(\mathbf{z} - E[\mathbf{z}])\right) \\ & \qquad \left(\mathbf{x} - E[\mathbf{x}] - \mathrm{Cov}[\mathbf{x}, \mathbf{z}]\mathrm{Cov}[\mathbf{z}]^{-1}(\mathbf{z} - E[\mathbf{z}])\right)^\mathsf{T}\Big] \\ =& \mathrm{Cov}[\mathbf{x}] - \mathrm{Cov}[\mathbf{x}, \mathbf{z}]\mathrm{Cov}[\mathbf{z}]^{-1}\mathrm{Cov}[\mathbf{x}, \mathbf{z}]^\mathsf{T}. \end{aligned} \tag{2.82}$$

The LMMSE equations (2.74) and (2.82) bears some resemblance to the Kalman filter formulas. In fact, we could have derived the Kalman filter as an example of LMMSE estimation by means of the orthogonality principle. We shall instead, however, study the Kalman filter in a more directly Bayesian setting, because this will make it easier to understand how the Kalman filter is used in sensor fusion applications such as target tracking and SLAM. For this purpose, the next chapter is entirely devoted to the multivariate Gaussian distribution.

2.7 References and chapter remarks

The aim of this chapter has largely been to provide a condensed summary of the first three chapters in [3]. Thus, if the reader is looking for further details, that would be a good place to start. For fundamentals of probability theory, the reader may consult a basic probability textbook such as [91], or go straight to the more comprehensive [70]. The proof for Bayes' rule for pdf's in Theorem 2.5.1 is based on a somewhat more cursorial proof found in [70]. To do anything with the measure theoretic approach to probability must students who follow TTK4250 would first need to become familiar with measure theory, which is rigorously explained in [31]. The best source on Bayesian estimation and decision theory known to the author is [29].

2.8 Exercises

Exercise 2.1 Let $Y = X_1 + \ldots + X_n$ be the sum of n independent random vectors. Show that the characteristic function of Y as given by (2.42) is equal to the product of the characteristic functions of $X_1, \ldots, X_n$. ■

Exercise 2.2 Let $X_1, \ldots, X_n$ be n i.i.d. random variables, each χ^2 distributed with k degrees of freedom. Show that $Y = X_1 + \ldots + X_n$ is χ^2 distributed with nk degrees of freedom. ■

Exercise 2.3 Let X be a Rayleigh distributed random variable with parameter σ^2 as defined on page 21. Show that X^2 is an exponential random variable with parameter $\lambda = 1/\sigma^2$. ■

Exercise 2.4 Prove the Lindeberg-Lévy version of the CLT. That is, prove Theorem 2.3.1. **Hint:** Use generating functions. ■

Exercise 2.5 Let $X_1, \ldots, X_n$ be n independent Bernoulli random variables with success probabilities $r_1, \ldots, r_n$, respectively.

a) What is distribution of $Y = \sum_{i=1}^{n} X_n$?
b) What is the expectation of Y?
c) What is the covariance of Y?

■

Exercise 2.6 Let X be a random variable with cumulative distribution function F. Show that the random variable $Y = F(X)$ is uniformly distributed over $[0,1]$. ■

Exercise 2.7 Show that the MAP classifier has the highest success rate of all possible classifiers for a Bayesian classification problem. ■

3. The multivariate Gaussian

The key construction that underlies the Kalman filter, and virtually all of sensor fusion, is the multivariate Gaussian, which generalizes the univariate Gaussian from Example (2.6). The distribution of a multivariate Gaussian RV is given by its expectation vector μ and symmetric positive definite covariance matrix $\mathbf{P}$, according to

$$\mathcal{N}(\mathbf{x};\mu,\mathbf{P}) = \frac{1}{(2\pi)^{\frac{n}{2}}|\mathbf{P}|^{\frac{1}{2}}}\exp\left(-\frac{1}{2}(\mathbf{x}-\mu)^{\mathsf{T}}\mathbf{P}^{-1}(\mathbf{x}-\mu)\right) \tag{3.1}$$

Recall that the univariate Gaussian pdf always looks like a bell curve whose peak location and spread are governed by its expectation and covariance, respectively. A two-dimensional Gaussian gives a similar bell surface, whose peak location is given by its expectation vector, and whose shape is given by its covariance matrix. This generalizes in the obvious manner to higher dimensions.

3.1 Quadratic forms and covariance ellipses

The exponent in (3.1) is a quadratic form in the variable $\mathbf{x}$. Let us more precisely define the quadratic form corresponding to (3.1) as

$$q(\mathbf{x}) = (\mathbf{x}-\mu)^{\mathsf{T}}\mathbf{P}^{-1}(\mathbf{x}-\mu). \tag{3.2}$$

The value of this function for a given $\mathbf{x}$ and μ is also known as the Mahalanobis distance between $\mathbf{x}$ and μ. In terms of this function we can write the logarithm of the Gaussian pdf as

$$\ln\mathcal{N}(\mathbf{x};\mu,\mathbf{P}) = c - \frac{1}{2}q(\mathbf{x}) \tag{3.3}$$

where c is a constant that is given by the normalization requirement, and therefore carries no information about $\mathbf{x}$. From this we make three very important observations.

First, this means that the curvature, or more precisely the Hessian, of $\ln\mathcal{N}(\mathbf{x};\mu,\mathbf{P})$ is constant. In fact, if $\mathcal{H}_{\mathbf{x}}$ denotes the Hessian, the following identity holds for any $\mathbf{x}$:

$$\mathcal{H}_{\mathbf{x}}\ln\mathcal{N}(\mathbf{x};\mu,\mathbf{P}) = -\mathbf{P}^{-1}. \tag{3.4}$$

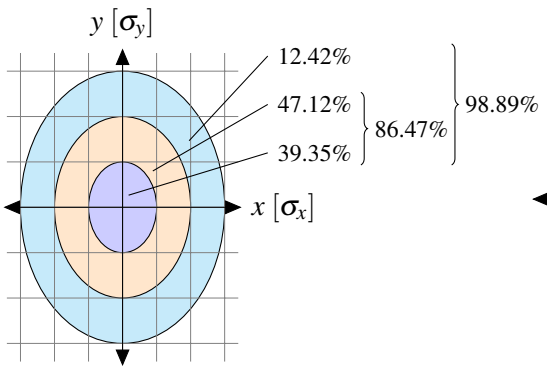
Second, the level curves for a 2-dimensional Gaussian are ellipses, since any expression of the form $Ax^2 + Bxy + Cy^2 = 1$ describes an ellipse in the $x-y$-plane. More generally, we get ellipsoids for a 3-dimensional Gaussian, and so on. For simplicity, we will just refer to these as covariance ellipses, irrespectively of the dimension. The axes of these ellipses can be found by means of eigenvalue decomposition of $\mathbf{P}$. If $(\lambda_i, \mathbf{e}_i)$ are the eigenvalues and eigenvectors of $\mathbf{P}$, the the 1σ ellipse of $\mathbf{P}$ has axes given by $\sqrt{\lambda_i}\mathbf{e}_i$, and the more general $g\sigma$-ellipse has axes given by $g\sqrt{\lambda_i}\mathbf{e}_i$. The area, or more generally hypervolume, within such an ellipse is given by

$$\frac{\pi^{n/2}}{\Gamma(n/2+1)} g^n \sqrt{|\mathbf{P}|}.$$

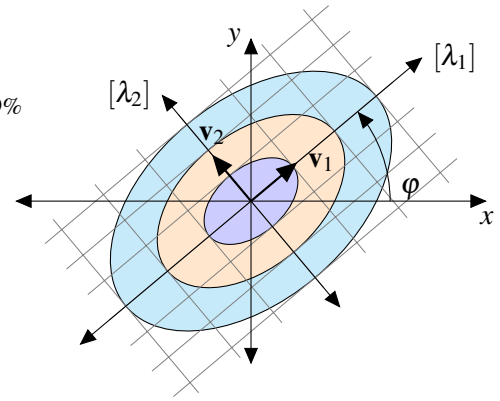
Third, it is evident that a Gaussian is entirely specified in terms of its quadratic form. Conversely, if we multiply a convex quadratic form in $\mathbf{x}$ by $\left(-\frac{1}{2}\right)$, and then exponentiate it, we get a function which is proportional to a particular Gaussian distribution in $\mathbf{x}$. The normalization requirement implies that this function cannot be proportional to any other pdf than this particular Gaussian. The consequence of this is that when we study how multivariate Gaussians behave under various operations (marginalization, conditioning, etc.) *we only need to study the quadratic forms*.

■ **Example 3.1 — Conditioning is entirely given by quadratic forms.** Let $p(x,y)$ be a bivariate Gaussian, and let $f(x,y)$ be its corresponding quadratic form. The conditional distribution of x given y is $p(x|y) = p(x,y)/p(y)$. We know that both $p(x|y)$, $p(x,y)$ and $p(y)$ are valid pdfs. We also know that $p(y)$ is a function of y alone. The dependency of $p(x|y)$ on x is therefore given by the quadratic form $f(x,y)$, this time treated as a function of x alone, with y fixed. Thus, we see that $p(x|y)$ also must be Gaussian, and that we can determine $p(x|y)$ from $p(x,y)$ just by studying quadratic forms. The exact details for how this is to be done are left for Theorem 3.2.3 in the next section. ■

The amount of probability within a $g\sigma$-ellipse decreases as the dimension n increases. This is because higher dimension provides more directions in which probability mass can be spread. to quantify this, the trick is to transform the Gaussian random vector into a scalar χ^2 random variable whose cdf can be written in terms of the error function or directly evaluated using, e.g., Matlab. More precisely, if $\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{P})$ it can be shown that $(\mathbf{x}-\mu)^\mathsf{T}\mathbf{P}^{-1}(\mathbf{x}-\mu)$ has a χ^2 distribution with n degrees of freedom (Exercise 3.4). The probability mass within 1, 2 and 3 σ for a bivariate Gaussian is illustrated in Figure 3.1.



(a) $\Sigma = \begin{bmatrix} \sigma_x^2 & 0 \\ 0 & \sigma_y^2 \end{bmatrix}$.



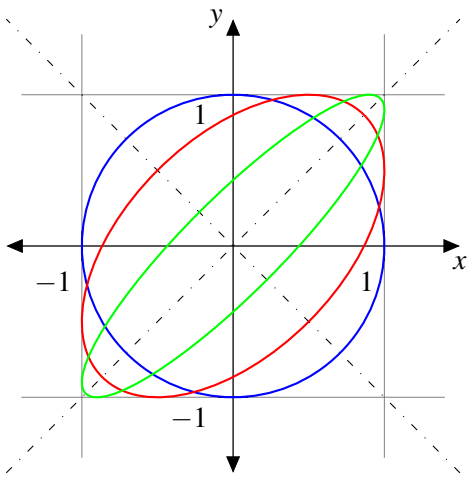
(b) $\Sigma = \mathbf{P}\Lambda\mathbf{P}^T$, $\mathbf{P} = \begin{bmatrix} \mathbf{v}_1 & \mathbf{v}_2 \end{bmatrix}$.

Figure 3.1: Two-dimensional Gaussian distribution. Probability of ellipses.

In the right hand side of Figure 3.1 we see how the similarity transform (also known as the eigendecomposition or spectral transform) of the covariance matrix yields an alternative coordinate

system, spanned by the eigenvectors, where all cross-correlations vanish. This can be useful for a variety of computational purposes. It is also possible to transform a correlated Gaussian vector to a correlation-free Gaussian vector by means of other decompositions such as the Cholesky factorization (see Example 3.4). We see that correlations make the covariance ellipses tilted. In the right hand side of Figure 3.1 there is a positive correlation between x and y. The ellipses would have been tilted the other way if it was negative.

■ **Example 3.2 — Correlations make the covariance ellipses narrower.** In Figure 3.2 we see the 1σ covariance ellipses of three bivariate Gaussians with unity marginal variance in the x- and y-directions. The top and bottom of all the covariance ellipses touch the -1 and $+1$ lines, and the same happens horizontally. However, the Gaussians with the highest cross-correlation a, have the narrowest covariance ellipses. This shows that the presence of correlations actually decrease uncertainty. For this reason, estimation methods such as the Kalman filter exploit correlations.



$$\begin{bmatrix} x \\ y \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 & a \\ a & 1 \end{bmatrix} \right)$$

$$\begin{bmatrix} x & y \end{bmatrix} \begin{bmatrix} 1 & a \\ a & 1 \end{bmatrix}^{-1} \begin{bmatrix} x \\ y \end{bmatrix} = 1$$

$a = 0.0$
$a = 0.5$
$a = 0.9$

Figure 3.2: For $a \in (-1,1)$, the semiaxes are $\sqrt{1+a}$ and $\sqrt{1-a}$ long and lie on the $y = x$ and $y = -x$ lines, respectively.

■

3.2 Rules for working with Gaussians

We will now establish several important results that make it much easier to work with with multivariate Gaussians than with any other kind of multivariate RVs. Based on the argument elaborated in Example 3.1, we will prove all these results by simply studying how the quadratic form in the exponent of (3.1) behaves.

Theorem 3.2.1 — Independence. Two random vectors $\mathbf{x}$ and $\mathbf{y}$ with probability density functions $\mathcal{N}(\mathbf{x};\mathbf{a},\mathbf{A})$ and $\mathcal{N}(\mathbf{y};\mathbf{b},\mathbf{B})$ are independent if and only if

$$p(\mathbf{x},\mathbf{y}) = p\left(\begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix}\right) = \mathcal{N}\left(\begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix}; \begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix}, \begin{bmatrix} \mathbf{A} & 0 \\ 0 & \mathbf{B} \end{bmatrix}\right). \tag{3.5}$$

Proof. We need to prove that zero covariance implies that $p(\mathbf{x},\mathbf{y}) = p(\mathbf{x})p(\mathbf{y})$ and vice versa. According to the inversion rule of Appendix P.1 we know that the inverse of a block-diagonal matrix is found by simply inverting the blocks. Thus, the quadratic form in the exponent of the

joint Gaussian becomes

$$\begin{bmatrix} \mathbf{x}-\mathbf{a} \\ \mathbf{y}-\mathbf{b} \end{bmatrix}^\mathsf{T} \begin{bmatrix} \mathbf{A} & 0 \\ 0 & \mathbf{B} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{x}-\mathbf{a} \\ \mathbf{y}-\mathbf{b} \end{bmatrix} = \begin{bmatrix} (\mathbf{x}-\mathbf{a})^\mathsf{T} & (\mathbf{y}-\mathbf{b})^\mathsf{T} \end{bmatrix} \begin{bmatrix} \mathbf{A}^{-1} & 0 \\ 0 & \mathbf{B}^{-1} \end{bmatrix} \begin{bmatrix} \mathbf{x}-\mathbf{a} \\ \mathbf{y}-\mathbf{b} \end{bmatrix}$$
$$= (\mathbf{x}-\mathbf{a})^\mathsf{T}\mathbf{A}^{-1}(\mathbf{x}-\mathbf{a}) + (\mathbf{y}-\mathbf{b})^\mathsf{T}\mathbf{B}^{-1}(\mathbf{y}-\mathbf{b})$$

which is a sum of the quadratic forms from the marginal distributions. Based on this, we see that zero covariance implies that $\ln p(\mathbf{x},\mathbf{y}) = \ln p(\mathbf{x}) + \ln p(\mathbf{y})$, which again implies that $p(\mathbf{x},\mathbf{y}) = p(\mathbf{x})p(\mathbf{y})$. The converse implications also follow. ■

This relationship between independence and zero covariance does not hold for arbitrary non-Gaussian random vectors.

There is a strong relationship between Gaussianity and linearity, to the extent that these concepts often are used interchangeably in the literature. If a Gaussian RV goes through a linear transform, the new RV is also Gaussian.

Theorem 3.2.2 — Linearity. If $p(\mathbf{x}) = \mathcal{N}(\mathbf{x}; \mathbf{a}, \mathbf{A})$ and $\mathbf{y} = \mathbf{F}\mathbf{x}$, then $p(\mathbf{y}) = \mathcal{N}(\mathbf{y}; \mathbf{F}\mathbf{a}, \mathbf{F}\mathbf{A}\mathbf{F}^\mathsf{T})$.

Proof. A general proof of this property can be constructed by means of the moment-generating function. In the special case that $\mathbf{F}$ is square and invertible, a simpler proof can be constructed by means of the nonlinear transformation formula in Theorem 2.4.1. We only sketch this proof here. Denote the original pdf by $g(\mathbf{x})$ and the pdf of the new RV by $h(\mathbf{y})$. We then have that

$$\begin{aligned} g(\mathbf{x}) &\propto \exp\left(-\frac{1}{2}(\mathbf{x}-\mathbf{a})^\mathsf{T}\mathbf{A}^{-1}(\mathbf{x}-\mathbf{a})\right) \\ h(\mathbf{y}) &\propto \exp\left(-\frac{1}{2}(\mathbf{F}^{-1}\mathbf{y}-\mathbf{a})^\mathsf{T}\mathbf{A}^{-1}(\mathbf{F}^{-1}\mathbf{y}-\mathbf{a})\right) \\ &= \exp\left(-\frac{1}{2}(\mathbf{F}^{-1}\mathbf{y}-\mathbf{a})^\mathsf{T}\mathbf{F}^\mathsf{T}(\mathbf{F}^\mathsf{T})^{-1}\mathbf{A}^{-1}\mathbf{F}^{-1}\mathbf{F}(\mathbf{F}^{-1}\mathbf{y}-\mathbf{a})\right) \\ &= \exp\left(-\frac{1}{2}(\mathbf{y}-\mathbf{F}\mathbf{a})^\mathsf{T}(\mathbf{F}^\mathsf{T})^{-1}\mathbf{A}^{-1}\mathbf{F}^{-1}(\mathbf{y}-\mathbf{F}\mathbf{a})\right). \end{aligned} \tag{3.6}$$

The theorem then follows from recognizing that $(\mathbf{F}^\mathsf{T})^{-1}\mathbf{A}^{-1}\mathbf{F}^{-1} = (\mathbf{F}\mathbf{A}\mathbf{F}^\mathsf{T})^{-1}$. ■

A related result is that if $\mathbf{x}$ has the distribution $\mathcal{N}(\mathbf{a},\mathbf{A})$, then the random vector $\mathbf{y} = \mathbf{x}+\mathbf{b}$ has the distribution $\mathcal{N}(\mathbf{a}+\mathbf{b},\mathbf{A})$. This follows almost trivially from the nonlinear transformation formula, because all we have to is shift $\mathbf{b}$ from the random vector slot to the expectation slot.

▪ **Example 3.3 — Sum of variances.** Let $x_1 \sim \mathcal{N}(0,\sigma^2)$ and $x_2 \sim \mathcal{N}(0,\sigma^2)$. Then it follows that $x = x_1 + x_2 \sim \mathcal{N}(0, 2\sigma^2)$. This follows from the theorem with $\mathbf{F} = [1,1]$, $\mathbf{x} = [x_1, x_2]^\mathsf{T}$ and $\mathbf{y} = x_3$. ▪

▪ **Example 3.4 — Cholesky decomposition of the covariance matrix.** Let $\mathbf{x} \sim \mathcal{N}(\mathbf{0},\mathbf{A})$. Since $\mathbf{A}$ is symmetric positive definite we know that it has a Cholesky factorization $\mathbf{L}$ so that $\mathbf{A} = \mathbf{L}\mathbf{L}^\mathsf{T}$. Let us then define the transformed RV $\mathbf{y} = \mathbf{L}^{-1}\mathbf{x}$. The expectation of $\mathbf{y}$ is then obviously zero as well. Its covariance becomes

$$\mathrm{Cov}[\mathbf{y}] = \mathbf{L}^{-1}\mathbf{A}(\mathbf{L}^{-1})^\mathsf{T} = \mathbf{L}^{-1}\mathbf{L}\mathbf{L}^\mathsf{T}(\mathbf{L}^{-1})^\mathsf{T} = \mathbf{I}. \tag{3.7}$$

The effect that the linear transformation $\mathbf{L}^{-1}$ has on $\mathbf{x}$ is often described as "whitening" or "prewhitening". This is a common and important operation, especially in signal processing, where $\mathbf{x}$ may be thought of as a long time sequence constituting a signal. By whitening a signal

we can perform operations on it (e.g, hypothesis tests) which require whiteness, i.e., independence between the samples. We can also use a reversal of the above argument to generate correlated Gaussian RVs. If we have a Gaussian RV $\mathbf{y} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, then the transformed RV $\mathbf{x} = \mathbf{L}\mathbf{y}$ will have the covariance

$$\mathrm{Cov}[\mathbf{x}] = \mathbf{L}\mathbf{L}^{\mathsf{T}} = \mathbf{A}. \tag{3.8}$$

In Matlab, we can for example draw `N` indepedendent random vectors from $\mathcal{N}(\mathtt{a}, \mathtt{A})$ using

$$\mathtt{x} = \mathtt{repmat}(\mathtt{a}, [1, \mathtt{N}]) + \mathtt{cholA'*randn}(\mathtt{n}, \mathtt{N}). \tag{3.9}$$

The transpose is required because Matlab's `chol` function calculates the upper Cholesky matrix by default, and not the lower Cholesky matrix. ∎

Marginalization and conditioning are frequent tasks that must be done with the multivariate Gaussian. In the moment-based representation of the Gaussian, the former is trivial, while the latter is somewhat more complicated.

Theorem 3.2.3 — Marginalization and conditioning. Let $\mathbf{x}$ and $\mathbf{y}$ have the joint distribution

$$p(\mathbf{x}, \mathbf{y}) = \mathcal{N}\left(\begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix}; \begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix}, \begin{bmatrix} \mathbf{P}_{xx} & \mathbf{P}_{xy} \\ \mathbf{P}_{xy}^{\mathsf{T}} & \mathbf{P}_{yy} \end{bmatrix}\right)$$

Then the marginal distribution of $\mathbf{y}$ is $p(\mathbf{y}) = \mathcal{N}(\mathbf{y}; \mathbf{b}, \mathbf{P}_{yy})$, and the conditional distribution pf $\mathbf{x}$ given $\mathbf{y}$ is $p(\mathbf{x}|\mathbf{y}) = \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_{x|y}, \mathbf{P}_{x|y})$ where $\boldsymbol{\mu}_{x|y} = \mathbf{a} + \mathbf{P}_{xy}\mathbf{P}_{yy}^{-1}(\mathbf{y} - \mathbf{b})$ and $\mathbf{P}_{x|y} = \mathbf{P}_{xx} - \mathbf{P}_{xy}\mathbf{P}_{yy}^{-1}\mathbf{P}_{xy}^{\mathsf{T}}$.

Proof. For simplicity, assume that $\mathbf{a}$ and $\mathbf{b}$ are zero. We use the matrix inversion rule (P.3) to decompose the inverse covariance matrix as follows:

$$\begin{bmatrix} \mathbf{P}_{xx} & \mathbf{P}_{xy} \\ \mathbf{P}_{xy}^{\mathsf{T}} & \mathbf{P}_{yy} \end{bmatrix}^{-1} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ -\mathbf{P}_{yy}^{-1}\mathbf{P}_{xy}^{\mathsf{T}} & \mathbf{I} \end{bmatrix} \begin{bmatrix} (\mathbf{P}_{xx} - \mathbf{P}_{xy}\mathbf{P}_{yy}^{-1}\mathbf{P}_{xy}^{\mathsf{T}})^{-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{P}_{yy}^{-1} \end{bmatrix} \begin{bmatrix} \mathbf{I} & -\mathbf{P}_{xy}\mathbf{P}_{yy}^{-1} \\ \mathbf{0} & \mathbf{I} \end{bmatrix}.$$

We insert this into the joint Gaussian, and obtain

$$p(\mathbf{x}, \mathbf{y}) \propto \exp\left(-\frac{1}{2}\begin{bmatrix} \mathbf{x} - \mathbf{P}_{xy}\mathbf{P}_{yy}^{-1}\mathbf{y} \\ \mathbf{y} \end{bmatrix}^{\mathsf{T}} \begin{bmatrix} (\mathbf{P}_{xx} - \mathbf{P}_{xy}\mathbf{P}_{yy}^{-1}\mathbf{P}_{xy}^{\mathsf{T}})^{-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{P}_{yy}^{-1} \end{bmatrix} \begin{bmatrix} \mathbf{x} - \mathbf{P}_{xy}\mathbf{P}_{yy}^{-1}\mathbf{y} \\ \mathbf{y} \end{bmatrix}\right).$$

Since the center matrix is diagonal, this becomes

$$p(\mathbf{x}, \mathbf{y}) \propto \exp\left(-\frac{1}{2}(\mathbf{x} - \mathbf{P}_{xy}\mathbf{P}_{yy}^{-1}\mathbf{y})^{\mathsf{T}}(\mathbf{P}_{xx} - \mathbf{P}_{xy}\mathbf{P}_{yy}^{-1}\mathbf{P}_{xy}^{\mathsf{T}})^{-1}(\mathbf{x} - \mathbf{P}_{xy}\mathbf{P}_{yy}^{-1}\mathbf{y})\right) \exp\left(-\frac{1}{2}\mathbf{y}^{\mathsf{T}}\mathbf{P}_{yy}^{-1}\mathbf{y}\right).$$

Both the desired results follow from inspection of this expression. We look at marginalization first. In this case, the goal is to find $p(\mathbf{y}) = \int p(\mathbf{x}, \mathbf{y})\mathrm{d}\mathbf{x}$. Irrespectively on the value of $\mathbf{y}$, the first exponential will describe a Gaussian in $\mathbf{x}$. For this reason, the integral over $\mathbf{x}$ eliminates this entire exponential, including its dependency on $\mathbf{y}$, and we are left with the second exponential. This proves the marginalization result. Then we look at conditioning. In this case, the goal is to find $p(\mathbf{x}|\mathbf{y}) = p(\mathbf{x}, \mathbf{y})/p(\mathbf{y})$. Carrying out this division amounts to removing the second exponential, so that we are left with only the first one. That is, the conditional density $p(\mathbf{x}|\mathbf{y})$ is a Gaussian with expectation $\mathbf{P}_{xy}\mathbf{P}_{yy}^{-1}\mathbf{y}$ and covariance $\mathbf{P}_{xx} - \mathbf{P}_{xy}\mathbf{P}_{yy}^{-1}\mathbf{P}_{xy}^{\mathsf{T}}$. Re-introducing the expectations $\mathbf{a}$ and $\mathbf{b}$ at their appropriate slots is straightforward, and concludes the proof. ∎

■ **Example 3.5 — Marginalization and conditioning.** When we condition on part of a Gaussian vector, we are essentially making a cut through the ellipse at the conditioning variables. Since the presence of correlations make the ellipse narrower, as seen in Example 3.2, we would expect the uncertainty that remains after conditioning to be smaller than the marginal uncertainty. This is of course also what the formulas in Theorem 3.2.3 tell us. Figure 3.3 illustrates this.

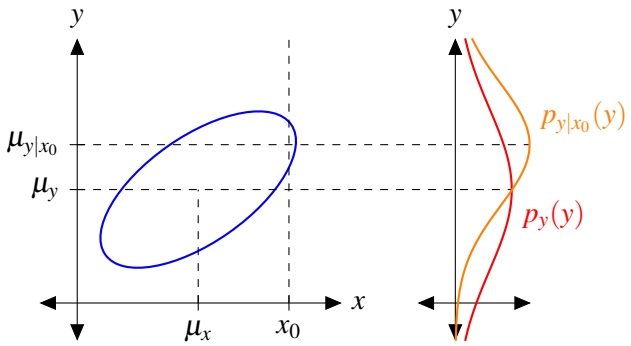


Figure 3.3: Two-dimensional Gaussian distribution. Marginal distribution (red) and conditional distribution (orange) for a value $x = x_0$.

■

3.3 The product identity

After having tasted some samples of the niceness of the multivariate Gaussian in Sections 3.2 - 3.4, it should not come as any surprise that the product of two Gaussians is another Gaussian. After all, a Gaussian is what we get when we exponentiate a negative definite quadratic form, and when we add two quadratic forms we get another quadratic form, whose exponential becomes a Gaussian after normalization. What is perhaps more surprising is that the normalization constant also becomes a Gaussian.

Recall that the expression for a Gaussian, $\mathcal{N}(\mathbf{x}; \mu, \mathbf{P})$, has two "slots" where vectors can be inserted, i.e., the argument slot and the expectation slot. What is particularly interesting is to see what happens when we have a product of two Gaussians, where $\mathbf{x}$ enters the argument slot in one of the Gaussians, and the expectation slot in the other Gaussian. This leads to the fundamental product identity.

Theorem 3.3.1 — The product identity. The following identity

$$\mathcal{N}(\mathbf{z}; \mathbf{H}\mathbf{x}, \mathbf{R})\mathcal{N}(\mathbf{x}; \bar{\mathbf{x}}, \bar{\mathbf{P}}) = \mathcal{N}(\mathbf{z}; \bar{\mathbf{z}}, \mathbf{S})\mathcal{N}(\mathbf{x}; \hat{\mathbf{x}}, \hat{\mathbf{P}}) \tag{3.10}$$

is true if the vectors and matrices involved are related according to

$$\begin{aligned}
\bar{\mathbf{z}} &= \mathbf{H}\bar{\mathbf{x}} \\
\hat{\mathbf{x}} &= \bar{\mathbf{x}} + \mathbf{W}(\mathbf{z} - \mathbf{H}\bar{\mathbf{x}}) \\
\mathbf{S} &= \mathbf{R} + \mathbf{H}\bar{\mathbf{P}}\mathbf{H}^\mathsf{T} \\
\hat{\mathbf{P}} &= (\mathbf{I} - \mathbf{W}\mathbf{H})\bar{\mathbf{P}} \\
\mathbf{W} &= \bar{\mathbf{P}}\mathbf{H}^\mathsf{T}\mathbf{S}^{-1}.
\end{aligned}$$

Proof. We can prove the product identity by showing that both the left-hand-side and the right-hand-side are two possible factorizations of a joint Gaussian $p(\mathbf{x}, \mathbf{z})$. In other words, we are going

to use the relationships

$$p(\mathbf{x},\mathbf{z}) = p(\mathbf{z}|\mathbf{x})p(\mathbf{x}) = p(\mathbf{x}|\mathbf{z})p(\mathbf{z}). \tag{3.11}$$

Step 1: Construct the joint density. If we start by looking at the first factorization in (3.11), we may simply define $p(\mathbf{z}|\mathbf{x})$ and $p(\mathbf{x})$ as identical to the two Gaussians on the left hand side of (3.10). This also defines $p(\mathbf{x},\mathbf{z})$ as identical to their product, i.e.,

$$p(\mathbf{z},\mathbf{x}) = \mathcal{N}(\mathbf{z};\mathbf{H}\mathbf{x},\mathbf{R})\mathcal{N}(\mathbf{x};\bar{\mathbf{x}},\bar{\mathbf{P}}). \tag{3.12}$$

Step 2: Manipulate the quadratic form. The quadratic form in (3.12) can be written as follows:

$$\begin{aligned}
&(\mathbf{x}-\bar{\mathbf{x}})^{\mathsf{T}}\bar{\mathbf{P}}^{-1}(\mathbf{x}-\bar{\mathbf{x}}) + (\mathbf{z}-\mathbf{H}\mathbf{x})^{\mathsf{T}}\mathbf{R}^{-1}(\mathbf{z}-\mathbf{H}\mathbf{x}) \\
&= \begin{bmatrix}\mathbf{x}-\bar{\mathbf{x}}\\ \mathbf{z}-\mathbf{H}\mathbf{x}\end{bmatrix}^{\mathsf{T}} \begin{bmatrix}\bar{\mathbf{P}}^{-1} & \mathbf{0}\\ \mathbf{0} & \mathbf{R}^{-1}\end{bmatrix} \begin{bmatrix}\mathbf{x}-\bar{\mathbf{x}}\\ \mathbf{z}-\mathbf{H}\mathbf{x}\end{bmatrix} \\
&= \begin{bmatrix}\mathbf{x}-\bar{\mathbf{x}}\\ \mathbf{z}-\mathbf{H}\bar{\mathbf{x}}\end{bmatrix}^{\mathsf{T}} \begin{bmatrix}\mathbf{I} & -\mathbf{H}^{\mathsf{T}}\\ \mathbf{0} & \mathbf{I}\end{bmatrix} \begin{bmatrix}\bar{\mathbf{P}}^{-1} & \mathbf{0}\\ \mathbf{0} & \mathbf{R}^{-1}\end{bmatrix} \begin{bmatrix}\mathbf{I} & \mathbf{0}\\ -\mathbf{H} & \mathbf{I}\end{bmatrix} \begin{bmatrix}\mathbf{x}-\bar{\mathbf{x}}\\ \mathbf{z}-\mathbf{H}\bar{\mathbf{x}}\end{bmatrix} \\
&= \begin{bmatrix}\mathbf{x}-\bar{\mathbf{x}}\\ \mathbf{z}-\mathbf{H}\bar{\mathbf{x}}\end{bmatrix}^{\mathsf{T}} \left(\begin{bmatrix}\mathbf{I} & \mathbf{H}\\ \mathbf{0} & \mathbf{I}\end{bmatrix} \begin{bmatrix}\bar{\mathbf{P}} & \mathbf{0}\\ \mathbf{0} & \mathbf{R}\end{bmatrix} \begin{bmatrix}\mathbf{I} & \mathbf{H}^{\mathsf{T}}\\ \mathbf{0} & \mathbf{I}\end{bmatrix} \right)^{-1} \begin{bmatrix}\mathbf{x}-\bar{\mathbf{x}}\\ \mathbf{z}-\mathbf{H}\bar{\mathbf{x}}\end{bmatrix} \\
&= \begin{bmatrix}\mathbf{x}-\bar{\mathbf{x}}\\ \mathbf{z}-\mathbf{H}\bar{\mathbf{x}}\end{bmatrix}^{\mathsf{T}} \begin{bmatrix}\bar{\mathbf{P}} & \bar{\mathbf{P}}\mathbf{H}^{\mathsf{T}}\\ \mathbf{H}\bar{\mathbf{P}} & \mathbf{H}\bar{\mathbf{P}}\mathbf{H}^{\mathsf{T}}+\mathbf{R}\end{bmatrix}^{-1} \begin{bmatrix}\mathbf{x}-\bar{\mathbf{x}}\\ \mathbf{z}-\mathbf{H}\bar{\mathbf{x}}\end{bmatrix}
\end{aligned} \tag{3.13}$$

The second equality in this development is the only one that is not straightforward. The background for this equality is that we need to remove $\mathbf{x}$ from the outer vector if we are going to arrive at the product identity. This is done by subjecting the outer vector to an appropriate linear transform. More precisely, this is achieved by

$$\begin{bmatrix}\mathbf{I} & \mathbf{0}\\ -\mathbf{H} & \mathbf{I}\end{bmatrix} \begin{bmatrix}\mathbf{x}-\bar{\mathbf{x}}\\ \mathbf{z}-\mathbf{H}\bar{\mathbf{x}}\end{bmatrix} = \begin{bmatrix}\mathbf{x}-\bar{\mathbf{x}}\\ -\mathbf{H}\mathbf{x}+\mathbf{H}\bar{\mathbf{x}}+\mathbf{x}-\mathbf{H}\bar{\mathbf{x}}\end{bmatrix} = \begin{bmatrix}\mathbf{x}-\bar{\mathbf{x}}\\ \mathbf{z}-\mathbf{H}\mathbf{x}\end{bmatrix} \tag{3.14}$$

In other words, we can replace the right-hand-side of (3.14) with the left-hand-side of (3.14), which is done in the third line of (3.13).

Step 3: Factorize into marginal and conditional. The last expression in (3.13) is the quadratic form of a Gaussian in $\mathbf{x}$ and $\mathbf{z}$, and since it encapsulates all dependency on $\mathbf{x}$ and $\mathbf{z}$, we can rest assured that this Gaussian is the joint density $p(\mathbf{x},\mathbf{z})$. The covariance matrix in (3.13) is not block-diagonal, and therefore some further work is needed to split the quadratic form into a sum of two separate quadratic forms, as we need to arrive at the product identity. However, if we can construct the alternative factorization $p(\mathbf{x},\mathbf{z}) = p(\mathbf{x}|\mathbf{z})p(\mathbf{z})$ we may perhaps achieve this. To find the quadratic forms corresponding to $p(\mathbf{z})$ and $p(\mathbf{x}|\mathbf{z})$ we use Theorem 3.2.3, which provided expressions for marginal and conditional Gaussians. If we compare with the entities involved in Theorem 3.2.3, we can make the following identifications

Role	In Theorem 3.2.3	In (3.13)
Conditioned RV	$\mathbf{x}$	$\mathbf{x}$
RV that we condition on	$\mathbf{y}$	$\mathbf{z}$
Expectation of conditioned RV	$\mathbf{a}$	$\bar{\mathbf{x}}$
Expectation of RV that we condition on	$\mathbf{b}$	$\mathbf{H}\bar{\mathbf{x}} = \bar{\mathbf{z}}$
Covariance of conditioned RV	$\mathbf{P}_{xx}$	$\bar{\mathbf{P}}$
Cross-covariance	$\mathbf{P}_{xy}$	$\bar{\mathbf{P}}\mathbf{H}^{\mathsf{T}}$
Covariance of RV that we condition on	$\mathbf{P}_{yy}$	$\mathbf{H}\bar{\mathbf{P}}\mathbf{H}^{\mathsf{T}}+\mathbf{R} = \mathbf{S}$

Based on this, we may furthermore identify $\boldsymbol{\mu}_{x|y}$ and $\mathbf{P}_{x|y}$ in Theorem 3.2.3 with the entities

$$\hat{\mathbf{x}} = \bar{\mathbf{x}} + \bar{\mathbf{P}}\mathbf{H}^\mathsf{T}(\mathbf{H}\bar{\mathbf{P}}\mathbf{H}^\mathsf{T} + \mathbf{R})^{-1}(\mathbf{z} - \mathbf{H}\bar{\mathbf{x}}) \tag{3.15}$$

$$\hat{\mathbf{P}} = \bar{\mathbf{P}} - \bar{\mathbf{P}}\mathbf{H}^\mathsf{T}(\mathbf{H}\bar{\mathbf{P}}\mathbf{H}^\mathsf{T} + \mathbf{R})^{-1}\mathbf{H}\bar{\mathbf{P}}. \tag{3.16}$$

We have thus specified all the entities involved in the quadratic forms corresponding to $p(\mathbf{z})$ and $p(\mathbf{x}|\mathbf{z})$, and it follows that

$$(\mathbf{x} - \bar{\mathbf{x}})^\mathsf{T}\bar{\mathbf{P}}^{-1}(\mathbf{x} - \bar{\mathbf{x}}) + (\mathbf{z} - \mathbf{H}\mathbf{x})^\mathsf{T}\mathbf{R}^{-1}(\mathbf{z} - \mathbf{H}\mathbf{x}) = (\mathbf{z} - \bar{\mathbf{z}})^\mathsf{T}\mathbf{S}^{-1}(\mathbf{z} - \bar{\mathbf{z}}) + (\mathbf{x} - \hat{\mathbf{x}})^\mathsf{T}\hat{\mathbf{P}}^{-1}(\mathbf{x} - \hat{\mathbf{x}})$$

In order to arrive at the final expressions in the theorem, we notice first that the term $\mathbf{W} = \bar{\mathbf{P}}\mathbf{H}^\mathsf{T}\mathbf{S}^{-1}$ is present in both (3.15) and (3.16). In the first of these, recognizing $\mathbf{W}$ leads to the expression $\hat{\mathbf{x}} = \bar{\mathbf{x}} + \mathbf{W}(\mathbf{z} - \mathbf{H}\bar{\mathbf{x}})$. In the second of these, recognizing $\mathbf{W}$ leads to $\hat{\mathbf{P}} = \bar{\mathbf{P}} - \mathbf{W}\mathbf{H}\bar{\mathbf{P}} = (\mathbf{I} - \mathbf{W}\mathbf{H})\bar{\mathbf{P}}$, and we have successfully separated (3.13) into the two quadratic forms on the right-hand-side of the product identity. ■

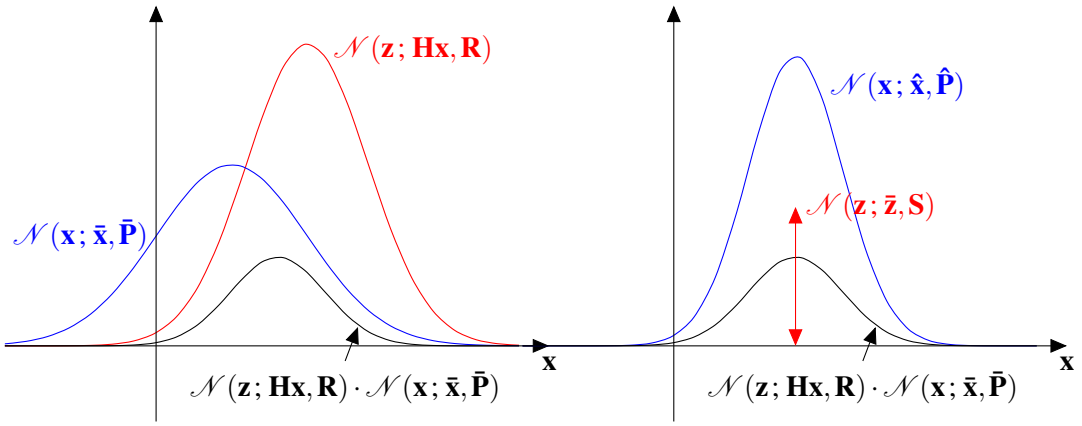


Figure 3.4: Illustration of the product identity.

3.4 The canonical form

As an alternative to the moment-based parameterization (3.1), the multivariate Gaussian can also parameterized in the canonical form, which in the multivariate case becomes

$$\mathcal{N}(\mathbf{x}; \mu, \mathbf{P}) = \exp\left(a + \boldsymbol{\eta}^\mathsf{T}\mathbf{x} - \frac{1}{2}\mathbf{x}^\mathsf{T}\boldsymbol{\Lambda}\mathbf{x}\right) \tag{3.17}$$

where

$$\boldsymbol{\Lambda} = \mathbf{P}^{-1} \tag{3.18}$$

$$\boldsymbol{\eta} = \boldsymbol{\Lambda}\mu \tag{3.19}$$

$$a = -(1/2)n\ln(2\pi) - \ln|\boldsymbol{\Lambda}| + \boldsymbol{\eta}^\mathsf{T}\boldsymbol{\Lambda}\boldsymbol{\eta}. \tag{3.20}$$

The entity $\boldsymbol{\eta}$ is sometimes referred to as an information state. The entity $\boldsymbol{\Lambda}$ is known as the information matrix or precision matrix. The term "information" refers to the fact that $\boldsymbol{\Lambda}$, being the inverse of $\mathbf{P}$, is a measure of the opposite of uncertainty.

We saw in the previous section that in the moment-based parametrization, marginalization was easy while conditioning was more complicated. In canonical form, it is opposite.

Theorem 3.4.1 — Marginalization and conditioning in canonical form. If $\mathbf{x}$ and $\mathbf{y}$ have the joint distribution

$$p(\mathbf{x},\mathbf{y}) \propto \exp\left(\begin{bmatrix} \boldsymbol{\eta}_a^\mathsf{T} & \boldsymbol{\eta}_b^\mathsf{T} \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix} - \frac{1}{2} \begin{bmatrix} \mathbf{x}^\mathsf{T} & \mathbf{y}^\mathsf{T} \end{bmatrix} \begin{bmatrix} \boldsymbol{\Lambda}_{xx} & \boldsymbol{\Lambda}_{xy} \\ \boldsymbol{\Lambda}_{xy}^\mathsf{T} & \boldsymbol{\Lambda}_{yy} \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix} \right) \tag{3.21}$$

then the marginal distribution of $\mathbf{y}$ is has potential vector $\boldsymbol{\eta}_* = \boldsymbol{\eta}_b - \boldsymbol{\Lambda}_{xy}^\mathsf{T}\boldsymbol{\Lambda}_{xx}^{-1}\boldsymbol{\eta}_a$ and information matrix $\boldsymbol{\Lambda}_* = \boldsymbol{\Lambda}_{yy} - \boldsymbol{\Lambda}_{xy}^\mathsf{T}\boldsymbol{\Lambda}_{xx}^{-1}\boldsymbol{\Lambda}_{xy}$, and the conditional distribution of $\mathbf{x}$ given $\mathbf{y}$ has potential vector $\boldsymbol{\eta}_{x|y} = \boldsymbol{\eta}_a - \boldsymbol{\Lambda}_{xy}\boldsymbol{\eta}_b$ and information matrix $\boldsymbol{\Lambda}_{x|y} = \boldsymbol{\Lambda}_{xx}$.

Proof. First we show the marginalization, and then we use this to show the conditioning. For the marginalized density, the information matrix must be the inverse of the corresponding marginalized covariance matrix. This means that

$$\begin{aligned}
\boldsymbol{\Lambda}_* =& \mathbf{P}_{yy}^{-1} \\
=& \left([\mathbf{0},\mathbf{I}] \begin{bmatrix} \mathbf{P}_{xx} & \mathbf{P}_{xy} \\ \mathbf{P}_{xy}^\mathsf{T} & \mathbf{P}_{yy} \end{bmatrix} \begin{bmatrix} \mathbf{0} \\ \mathbf{I} \end{bmatrix} \right)^{-1} = \left([\mathbf{0},\mathbf{I}] \begin{bmatrix} \boldsymbol{\Lambda}_{xx} & \boldsymbol{\Lambda}_{xy} \\ \boldsymbol{\Lambda}_{xy}^\mathsf{T} & \boldsymbol{\Lambda}_{yy} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{0} \\ \mathbf{I} \end{bmatrix} \right)^{-1} \\
=& \left([\mathbf{0},\mathbf{I}] \begin{bmatrix} \cdots & \cdots \\ \cdots & (\boldsymbol{\Lambda}_{yy} - \boldsymbol{\Lambda}_{xy}^\mathsf{T}\boldsymbol{\Lambda}_{xx}^{-1}\boldsymbol{\Lambda}_{xy})^{-1} \end{bmatrix} \begin{bmatrix} \mathbf{0} \\ \mathbf{I} \end{bmatrix} \right)^{-1} \\
=& \boldsymbol{\Lambda}_{yy} - \boldsymbol{\Lambda}_{xy}^\mathsf{T}\boldsymbol{\Lambda}_{xx}^{-1}\boldsymbol{\Lambda}_{xy}.
\end{aligned} \tag{3.22}$$

We have inverted the $\boldsymbol{\Lambda}$ matrix by means of the inverse block matrix formula from Appendix P.1. Since the inverted matrix is to be pre- and post-multiplied by the matrix $[\mathbf{0},\mathbf{I}]$ we have simply written "…" instead of spelling out the irrelevant blocks. In order to express the marginalized potential, we first express the marginalized expectation in terms of the joint potential vector:

$$\mathbf{b} = [\mathbf{0},\mathbf{I}] \begin{bmatrix} \boldsymbol{\Lambda}_{xx} & \boldsymbol{\Lambda}_{xy} \\ \boldsymbol{\Lambda}_{xy}^\mathsf{T} & \boldsymbol{\Lambda}_{yy} \end{bmatrix}^{-1} \begin{bmatrix} \boldsymbol{\eta}_a \\ \boldsymbol{\eta}_b \end{bmatrix} \tag{3.23}$$

The marginalized potential vector can then be found as

$$\boldsymbol{\eta}_* = \boldsymbol{\Lambda}_*\mathbf{b} = \boldsymbol{\Lambda}_*(\boldsymbol{\Lambda}_*^{-1}\boldsymbol{\Lambda}_{yy} - \boldsymbol{\Lambda}_*^{-1}\boldsymbol{\Lambda}_{xy}^\mathsf{T}\boldsymbol{\Lambda}_{xx}^{-1}\boldsymbol{\eta}_a) = \boldsymbol{\eta}_b - \boldsymbol{\Lambda}_{xy}^\mathsf{T}\boldsymbol{\Lambda}_{xx}^{-1}\boldsymbol{\eta}_a. \tag{3.24}$$

The conditional Gaussian $p(\mathbf{x}|\mathbf{y})$ can then be found according to

$$\begin{aligned}
p(\mathbf{x}|\mathbf{y}) =& \frac{p(\mathbf{x},\mathbf{y})}{p(\mathbf{y})} \\
\propto& \exp\left(\boldsymbol{\eta}_a^\mathsf{T}\mathbf{x} + \boldsymbol{\eta}_b^\mathsf{T}\mathbf{y} - \frac{1}{2}\left(\mathbf{x}^\mathsf{T}\boldsymbol{\Lambda}_{xx}\mathbf{x} + \mathbf{x}^\mathsf{T}\boldsymbol{\Lambda}_{xy}\mathbf{y} + \mathbf{y}^\mathsf{T}\boldsymbol{\Lambda}_{xy}^\mathsf{T}\mathbf{x} + \mathbf{y}^\mathsf{T}\boldsymbol{\Lambda}_{yy}\mathbf{y} \right) \right. \\
& \left. - \left(\boldsymbol{\eta}_b - \boldsymbol{\Lambda}_{xy}^\mathsf{T}\boldsymbol{\Lambda}_{xx}^{-1}\boldsymbol{\eta}_a \right)^\mathsf{T}\mathbf{y} - \frac{1}{2}\mathbf{y}^\mathsf{T}\left(\boldsymbol{\Lambda}_{yy} - \boldsymbol{\Lambda}_{xy}^\mathsf{T}\boldsymbol{\Lambda}_{xx}^{-1}\boldsymbol{\Lambda}_{xy} \right)\mathbf{y} \right) \\
\propto& \exp\left(\boldsymbol{\eta}_a^\mathsf{T}\mathbf{x} - \mathbf{y}^\mathsf{T}\boldsymbol{\Lambda}_{xy}\mathbf{x} - \frac{1}{2}\mathbf{x}^\mathsf{T}\boldsymbol{\Lambda}_{xx}\mathbf{x} \right)
\end{aligned} \tag{3.25}$$

The last proportionality is obtained by discarding all terms that do not contain $\mathbf{x}$, as they provide no information about the shape of the conditional distribution of $\mathbf{x}$. We see that $\boldsymbol{\eta}_a^\mathsf{T} - \mathbf{y}^\mathsf{T}\boldsymbol{\Lambda}_{xy}$ plays the role of the potential vector, while $\boldsymbol{\Lambda}_{xx}$ plays the role of the information matrix in the conditional distribution. ■

The canonical form is useful for several reasons. First, as shown in Theorem 3.4.1, conditioning is in general easier to do in canonical form than in the moment-based form. Since estimation often boils down to calcuating the marginal density of state, given the data, this is not unimportant. Furthermore, as already pointed out in (3.4), the information matrix equals the curvature of the logarithm of the Gaussian.

3.5 References and chapter remarks

The multivariate Gaussian is so important that extensive treatments can be found in many textbooks. From a statistical perspective, a standard reference is [48]. Readers more inclined towards machine learning, may prefer machine learning textbooks such as [63] or [9]. Sensor fusion textbooks such as [3] and [42] also contain all the standard stuff. The main purpose for the treatment in this book has been to present the most important results in manner as condensed and pedagogical as possible, while still provide rigorous proofs for all the results. It is therefore important to be aware that there is much more to read in the mentioned textbooks.

The strong focus on understanding the Gaussian as the exponential of a quadratic form is somewhat original to this book, although a similar perspective can also be found in [9]. The proof of Theorem 3.2.3 (marginalization and conditioning of Gaussians in moment-form) is due to Gary B. Huang [45], and goes along the lines of the proof in [63]. An alternative proof, formulated in terms of random variables, can be found in [48]. A third proof, which uses completion of the square, is found in [78]. The proof of the product identity follows the structure of the proof in [86]. Alternative proofs can be found in [75] and [59].

3.6 Exercises

Exercise 3.1 Let $\mathbf{A}$ and $\mathbf{B}$ be symmetric and invertible matrices of the same dimension. Show that

$$\mathbf{A} - \mathbf{A}(\mathbf{A}+\mathbf{B})^{-1}\mathbf{A} = (\mathbf{A}^{-1}+\mathbf{B}^{-1})^{-1}.$$

Hint: The equality holds if and only if the inverse of the right-hand-side times the left-hand-side equals the identity matrix. ∎

Exercise 3.2 Let $\mathbf{z} = \mathbf{H}\mathbf{x} + \mathbf{w}$ where $\mathbf{x}$ and $\mathbf{w}$ are independent random vectors distributed according to $\mathcal{N}(\mathbf{x}; |\bar{\mathbf{x}}, \mathbf{P})$ and $\mathcal{N}(\mathbf{w}; |\mathbf{0}, \mathbf{R})$. Show that the joint distribution of $\mathbf{x}$ and $\mathbf{z}$ is

$$\mathcal{N}\left(\begin{bmatrix}\mathbf{x}\\ \mathbf{z}\end{bmatrix}; \begin{bmatrix}\bar{\mathbf{x}}\\ \mathbf{0}\end{bmatrix}, \begin{bmatrix}\mathbf{P} & \mathbf{P}\mathbf{H}^{\mathsf{T}}\\ \mathbf{H}\mathbf{P} & \mathbf{H}\mathbf{P}\mathbf{H}^{\mathsf{T}}+\mathbf{R}\end{bmatrix}\right).$$

∎

Exercise 3.3 Derive the covariance formula $\hat{\mathbf{P}}^{-1} = \mathbf{H}^{\mathsf{T}}\mathbf{R}^{-1}\mathbf{H} + \bar{\mathbf{P}}^{-1}$. ∎

Exercise 3.4 Let $\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{P})$ be an n-dimensional RV. Show that $(\mathbf{x}-\mu)^{\mathsf{T}}\mathbf{P}^{-1}(\mathbf{x}-\mu)$ has a χ^2 distribution with n degrees of freedom.
Hint: You can build on relevant results from Examples 2.9, 2.10 and 3.4 as well as Theorem 3.2.2 and Exercise 2.2. ∎

4. The Kalman Filter From a Bayesian Perspective

Typical applications of sensor fusion involve temporal processing of sensor data. We are dealing with a dynamical system whose state we want to estimate as measurements arrive. For example, in a dynamical position system, we may receive measurements of position and orientation, and want to estimate the full state vector including position, orientation, velocities and bias terms for unmodeled dynamics, every time a measurement is received. This is known as state estimation. It is also commonly referred to as filtering in the estimation literature. As mentioned in Chapter 1 we can solve address state estimation with techniques based on both deterministic and probabilistic theory. On the latter is a focus here.

Having established a probabilistic framework in Chapters 2 - 3, we formulate the probabilistic filtering problem in relatively general terms in Section 4.1. After this is done, we invoke the common assumptions of Gaussianity and linearity in Section 4.2. This leads directly to the Kalman filter. In the subsequent sections we discuss how we can tune the Kalman filter according to knowledge about the noise processes.

4.1 The Bayes filter

In the filtering problem, the state of a stochastic dynamic system is to be estimated from a series of noisy measurements. By definition of the concept of a state vector, all information that describes the system at a given time is contained in the state vector.

■ **Example 4.1** In the most common kinematic model for target tracking, the so-called constant velocity model (CV model), the state vector contains only position and velocity. For a two-dimensional scenario, the state vector then becomes

$$\mathbf{x} = \begin{bmatrix} \rho_x \\ \rho_y \\ \mathbf{v}_x \\ \mathbf{v}_y \end{bmatrix} \tag{4.1}$$

where ρ_x is the position of the target along the x-coordinate etc. This entails an assumption that any

knowledge about, e.g., accelerations, previously or at this time, would not provide any additional information about how the target is going to move in the future. ∎

The filtering problem is formulated in terms of two models. The first model is known as the Markov model, kinematic prior or plant model, and concerns how the state evolves in time. This model can be specified as $p(\mathbf{x}_k \,|\, \mathbf{x}_{k-1})$. The second model is known as the measurement model or likelihood, and concerns how the measurements that we actually observe are related to the state vector. This model can be specified as $p(\mathbf{z}_k \,|\, \mathbf{x}_k)$.

A couple of independence assumptions underlie this way of treating the filtering problem. For the given specification of the plant model to make sense, the Markov property must hold:

$$p(\mathbf{x}_k \,|\, \mathbf{x}_1, \ldots, \mathbf{x}_{k-2}, \mathbf{x}_{k-1}, \mathbf{z}_1, \ldots, \mathbf{z}_{k-2}, \mathbf{z}_{k-1}) = p(\mathbf{x}_k \,|\, \mathbf{x}_{k-1}). \tag{4.2}$$

In other words, $\mathbf{x}_k$ should be independent of $\mathbf{x}_0, \ldots, \mathbf{x}_{k-1}$ when $\mathbf{x}_k$ is given. A similar requirement is assumed to hold for the measurements:

$$p(\mathbf{z}_k \,|\, \mathbf{x}_1, \ldots, \mathbf{x}_{k-2}, \mathbf{x}_{k-1}, \mathbf{x}_k, \mathbf{z}_1, \ldots, \mathbf{z}_{k-2}, \mathbf{z}_{k-1}) = p(\mathbf{z}_k \,|\, \mathbf{x}_k). \tag{4.3}$$

We can visualize this structure using the graphical model in Figure 4.1. The independence assumptions manifest themselves as follows: The only node that affects $\mathbf{x}_{k+1}$ is $\mathbf{x}_k$, and the only node that affects $\mathbf{z}_k$ is also $\mathbf{x}_k$. Every horizontal arrow connects two state nodes at different times according to the process model. Every vertical arrow connects a measurement node with the corresponding state node according to the measurement model.

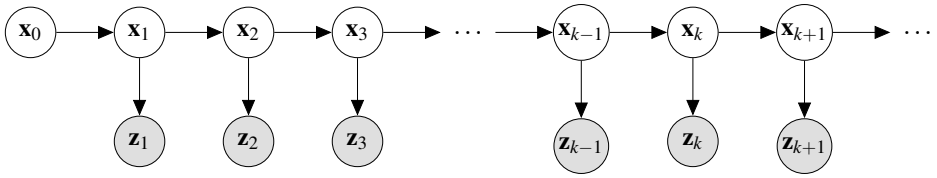


Figure 4.1: The probabilistic graphical model that underlies recursive Bayesian estimation.

Now that we have specified the filtering problem, it remains to solve it. In the Bayesian approach, we consider the posterior distribution $p(\mathbf{x}_k \,|\, \mathbf{z}_{1:k})$ as the solution to the filtering problem. The notation $\mathbf{z}_{1:k}$ signifies the sequence of all the measurements $\mathbf{z}_1, \ldots, \mathbf{z}_k$. The posterior distribution contains all the information that we have available about $\mathbf{x}_k$ conditional on these measurements. Once we have the posterior distribution specified, we can calculate various estimates of $\mathbf{x}_k$ based on MAP estimation, MMSE estimation etc.

The filtering is done in a cyclic manner, which consists of a prediction step and an update step. The prediction step, also known as the Chapman-Kolmogorov equation, follows from the total probability theorem and is given by

$$p(\mathbf{x}_k|\mathbf{z}_{1:k-1}) \;=\; \int p(\mathbf{x}_k, \mathbf{x}_{k-1} \,|\, \mathbf{z}_{1:k-1}) \mathrm{d}\mathbf{x}_{k-1} = \int p(\mathbf{x}_k|\mathbf{x}_{k-1}) p(\mathbf{x}_{k-1}|\mathbf{z}_{1:k-1}) \mathrm{d}\mathbf{x}_{k-1}. \tag{4.4}$$

The update step follows from Bayes' rule and is given by

$$p(\mathbf{x}_k|\mathbf{z}_{1:k}) \;=\; \frac{p(\mathbf{z}_k|\mathbf{x}_k) p(\mathbf{x}_k|\mathbf{z}_{1:k-1})}{p(\mathbf{z}_k \,|\, \mathbf{z}_{1:k-1})} \propto p(\mathbf{z}_k|\mathbf{x}_k) p(\mathbf{x}_k|\mathbf{z}_{1:k-1}). \tag{4.5}$$

Equations (4.4) and (4.5) are together known as the Bayes filter. These equations are the foundations for everything that follows in this book.

4.2 The Kalman filter

We cannot expect to find closed-form solutions to the Bayes filter in general. Most candidates for the pdfs $p(\mathbf{x}_k|\mathbf{x}_{k-1})$ and $p(\mathbf{x}_{k-1}|\mathbf{z}_{1:k-1})$ will not lead to any closed-form expression for the Chapman-Kolmogorov integral. Furthermore, when we multiply the pdfs $p(\mathbf{z}_k|\mathbf{x}_k)$ and $p(\mathbf{x}_k|\mathbf{z}_{1:k-1})$ in the Bayes formula, we have no guarantee that the normalized product will be recognizable as a known pdf. So, are there any circumstances under which we are guaranteed to get a closed-form solution?

The answer to this question is that a closed-form solution exist when both the Markov model and the likelihood are Gaussian and linear, and the initial density is Gaussian as well. That is, these models must be of the forms

$$\begin{aligned} p(\mathbf{x}_k|\mathbf{x}_{k-1}) &= \mathcal{N}(\mathbf{x}_k\,;\mathbf{F}\mathbf{x}_{k-1},\mathbf{Q}) \\ p(\mathbf{z}_k|\mathbf{x}_k) &= \mathcal{N}(\mathbf{z}_k\,;\mathbf{H}\mathbf{x}_k,\mathbf{R}) \\ p(\mathbf{x}_0) &= \mathcal{N}(\mathbf{x}_0\,;\hat{\mathbf{x}}_0,\mathbf{P}_0). \end{aligned} \tag{4.6}$$

This is often written in the equivalent form

$$\mathbf{x}_k = \mathbf{F}\mathbf{x}_{k-1} + \mathbf{v}_k, \qquad \mathbf{v}_k \sim \mathcal{N}(0,\mathbf{Q}) \tag{4.7}$$

$$\mathbf{z}_k = \mathbf{H}\mathbf{x}_k + \mathbf{w}_k, \qquad \mathbf{w}_k \sim \mathcal{N}(0,\mathbf{R}) \tag{4.8}$$

$$\mathbf{x}_0 \sim \mathcal{N}(\hat{\mathbf{x}}_0,\mathbf{P}_0). \tag{4.9}$$

which is more similar to the conventional state space formulation of linear systems in deterministic systems theory. Some remarks on notation are in order. The matrix $\mathbf{F}$ is known as the transition matrix while the matrix $\mathbf{H}$ is known as the measurement matrix. The symmetric positive definite matrices $\mathbf{Q}$ and $\mathbf{R}$ govern the statistical properties of the plant noise $\mathbf{v}_k$ and measurement noise $\mathbf{w}_k$, respectively. All noise vectors are assumed mutually independent. This follows from the Markov assumptions (4.2) and (4.3). The fact that $\mathbf{v}_k$ is independent of $\mathbf{v}_l$ for all $l \neq k$ is referred to as whiteness. Notice that whiteness does not imply Gaussianity, nor does Gaussianity imply whiteness.

Under these assumptions, the Kalman filter is an optimal solution to the estimation problem. It consists of the expressions provided in Algorithm 1:

Algorithm 1 The Kalman filter

procedure KF($\hat{\mathbf{x}}_{k-1}$, $\mathbf{P}_{k-1}$, $\mathbf{z}_k$)
- $\hat{\mathbf{x}}_{k|k-1} \leftarrow \mathbf{F}\hat{\mathbf{x}}_{k-1}$ ▷ The predicted state estimate
- $\mathbf{P}_{k|k-1} \leftarrow \mathbf{F}\mathbf{P}_{k-1}\mathbf{F}^\mathsf{T} + \mathbf{Q}$ ▷ The predicted covariance
- $\hat{\mathbf{z}}_{k|k-1} \leftarrow \mathbf{H}\hat{\mathbf{x}}_{k|k-1}$ ▷ The predicted measurement
- $\boldsymbol{\nu}_k \leftarrow \mathbf{z}_k - \hat{\mathbf{z}}_{k|k-1}$ ▷ The innovation
- $\mathbf{S}_k \leftarrow \mathbf{H}\mathbf{P}_{k|k-1}\mathbf{H}^\mathsf{T} + \mathbf{R}$ ▷ The innovation covariance
- $\mathbf{W}_k \leftarrow \mathbf{P}_{k|k-1}\mathbf{H}^\mathsf{T}\mathbf{S}_k^{-1}$ ▷ The Kalman gain
- $\hat{\mathbf{x}}_k \leftarrow \hat{\mathbf{x}}_{k|k-1} + \mathbf{W}_k\boldsymbol{\nu}_k$ ▷ The posterior state estimate
- $\mathbf{P}_k \leftarrow (\mathbf{I} - \mathbf{W}_k\mathbf{H})\mathbf{P}_{k|k-1}$ ▷ The posterior covariance
- **return** $\hat{\mathbf{x}}_k$, $\mathbf{P}_k$, $\hat{\mathbf{x}}_{k|k-1}$, $\mathbf{P}_{k|k-1}$, $\mathbf{S}_k$

end procedure

It is important to be familiar with all the equations and terminology in Algorithm 1. Keep in mind that some of the entities have different equivalent expressions, of which only one is given in Algorithm 1. In particular, the posterior covariance $\mathbf{P}_k$ can be written in more lengthy forms with

better numerical properties, such as the Joseph form

$$\mathbf{P}_k = (\mathbf{I} - \mathbf{W}_k\mathbf{H})\mathbf{P}_{k|k-1}(\mathbf{I} - \mathbf{W}_k\mathbf{H})^\mathsf{T} + \mathbf{W}\mathbf{R}\mathbf{W}^\mathsf{T}. \tag{4.10}$$

Recall that our aim is to obtain expressions for the predicted pdf $p(\mathbf{x}_k|\mathbf{z}_{1:k-1})$ and the posterior pdf $p(\mathbf{x}_k|\mathbf{z}_{1:k})$. The Kalman filter provides such expressions, and the relationship is explained in Theorems 4.2.1 and 4.2.1.

Theorem 4.2.1 — The Kalman filter prediction. If the posterior density at the previous time step is $p(\mathbf{x}_{k-1}\,|\,\mathbf{z}_{1:k-1}) = \mathcal{N}(\mathbf{x}_{k-1},;\hat{\mathbf{x}}_{k-1},\mathbf{P}_{k-1})$ and the Markov model is given as in (4.6), then the predicted density is $p(\mathbf{x}_k\,|\,\mathbf{z}_{1:k-1}) = \mathcal{N}(\mathbf{x}_k,;\hat{\mathbf{x}}_{k|k-1},\mathbf{P}_{k|k-1})$ where $\hat{\mathbf{x}}_{k|k-1}$ and $\mathbf{P}_{k|k-1}$ are as given in Algorithm 1.

Proof. We prove this by means of the fundamental product identity that was derived in Section 3.3. The predicted density is given by

$$\begin{aligned}
p(\mathbf{x}_k|\mathbf{z}_{1:k-1}) &= \int p(\mathbf{x}_k|\mathbf{x}_{k-1})p(\mathbf{x}_{k-1}|\mathbf{z}_{1:k-1})\mathrm{d}\mathbf{x}_{k-1} \\
&= \int \mathcal{N}(\mathbf{x}_k\,;\mathbf{F}\mathbf{x}_{k-1},Q)\mathcal{N}(\mathbf{x}_{k-1}\,;\hat{\mathbf{x}}_{k-1},\mathbf{P}_{k-1})\mathrm{d}\mathbf{x}_{k-1} \\
&= \mathcal{N}(\mathbf{x}_k\,;\mathbf{F}\hat{\mathbf{x}}_{k-1},\mathbf{F}\mathbf{P}_{k-1}\mathbf{F}^\mathsf{T}+Q) \\
&\quad \cdot \int \mathcal{N}(\mathbf{x}_{k-1}\,;\text{ some vector }\,,\text{ some covariance matrix }\,)\mathrm{d}\mathbf{x}_{k-1} \\
&= \mathcal{N}(\mathbf{x}_k\,;\hat{\mathbf{x}}_{k|k-1},\mathbf{P}_{k|k-1}).
\end{aligned} \tag{4.11}$$

where $\hat{\mathbf{x}}_{k|k-1}$ and $\mathbf{P}_{k|k-1}$ are as given in the theorem. ■

Theorem 4.2.2 — The Kalman filter update. If the predicted density at the current time step is $p(\mathbf{x}_k\,|\,\mathbf{z}_{1:k-1}) = \mathcal{N}(\mathbf{x}_{k-1},;\hat{\mathbf{x}}_{k|k-1},\mathbf{P}_{k|k-1})$ and the likelihood is given as in (4.6), then the posterior density is $p(\mathbf{x}_k\,|\,\mathbf{z}_{1:k}) = \mathcal{N}(\mathbf{x}_k,;\hat{\mathbf{x}}_k,\mathbf{P}_k)$ where $\hat{\mathbf{x}}_k$ and $\mathbf{P}_k$ are as given in Algorithm 1.

Proof. Again the proof is an application of the fundamental product identity. The posterior density is given by

$$\begin{aligned}
p(\mathbf{x}_k|\mathbf{z}_{1:k}) &\propto p(\mathbf{z}_k|\mathbf{x}_k)\,p(\mathbf{x}_k|\mathbf{z}_{1:k-1}) \\
&= \mathcal{N}(\mathbf{z}_k\,;\mathbf{H}\mathbf{x}_k,\mathbf{R})\,\mathcal{N}(\mathbf{x}_k\,;\hat{\mathbf{x}}_{k|k-1},\mathbf{P}_{k|k-1}) \\
&= \mathcal{N}(\mathbf{z}_k\,;\mathbf{H}\hat{\mathbf{x}}_{k|k-1},\mathbf{H}\mathbf{P}_{k|k-1}\mathbf{H}^\mathsf{T}+\mathbf{R})\mathcal{N}(\mathbf{x}_k\,;\hat{\mathbf{x}}_k,\mathbf{P}_k) \\
&\propto \mathcal{N}(\mathbf{x}_k\,;\hat{\mathbf{x}}_k,\mathbf{P}_k).
\end{aligned} \tag{4.12}$$

where $\hat{\mathbf{x}}_k$ and $\mathbf{P}_k$ are as given in the theorem. ■

This way of understanding the Kalman filter has both benefits and disadvantages compared to other approaches. As the reader hopefully can agree, the proof of the Kalman filter becomes quite trivial once the product identity is accepted as a premise. Proving the product identity was, however, non-trivial. When we later will extend the Kalman filter to tracking methods that operate in scenarios with multiple targets and clutter, the product identity will play an important role, so it is important to be familiar with it. Also, working directly in terms of Gaussian distributions, instead of only with expectation vectors and covariance matrices, highlights the relationship between the Kalman filter and the optimal Bayes filter. A disadvantage of this approach is that it fails to convey that the Kalman filter also has some interesting optimality properties when the noise is non-Gaussian, but zero-mean with a given variance, which Section 2.6.5 on LMMSE estimation

hinted at. Neither does it tell us anything about the stability of the Kalman filter from a control theoretic perspective.

We see that the design of a Kalman filter largely consists of choosing the matrices $\mathbf{F}$, $\mathbf{H}$, $\mathbf{Q}$ and $\mathbf{R}$. While $\mathbf{F}$ and $\mathbf{H}$ typically is given by the model, some degree of tuning is often required to set appropriate values in the covariance matrices $\mathbf{Q}$ and $\mathbf{R}$. In the remainder of this chapter we will discuss this in greater detail. Since the plant model in many tracking problems is most naturally specified in continuous time, the determination of $\mathbf{Q}$ is intimately linked with discretization of continuous time stochastic systems. For this reason, we need to take a closer look at the theory of stochastic processes.

4.3 Stochastic processes

We can view a stochastic process as an infinite-dimensional random variable. Recall that the possible outcomes of a scalar random variable are different numbers in $\mathbb{R}$. Similarly, the possible outcomes of a random vector (i.e., vector valued random variable) are different vectors in $\mathbb{R}^n$. For a (vector valued) stochastic process, the possible outcomes are different *functions* of the form $\mathbf{x}(t) : \mathbb{R} \to \mathbb{R}^n$. That is, a stochastic process describes a random function of time. The reader is strongly encouraged to spend some time digesting the following examples to get some tastes of the different structures that stochastic processes may have.

■ **Example 4.2 — The Wiener process.** This is the grandfather of most of the interesting stochastic processes. We first define a stochastic process $x(t)$ by

$$x(nT) = \sum_{i=1}^{n} x_i \tag{4.13}$$

where x_i $(i = 1, \ldots, n)$ are i.i.d. random variables with expectation 0 and variance T. The Wiener process $w(t)$ can then be defined as the limit

$$b(t) = \lim_{T \to 0} x(t). \tag{4.14}$$

Notice that we only need to specify the expectation and variance of the variables x_i, while their exact distribution can be arbitrary as long as they are i.i.d.. This is because of the central limit theorem. In the limit $T \to 0$ the number of i.i.d. random variables x_i that contribute to $b(t)$ will go towards infinity. Then the central limit theorem guarantees that the distribution of $b(t)$ is Gaussian. Furthermore, it is easy to determine the expectation and covariance of this Gaussian, for all values of t. Its expectation is zero, because the expectation of a sum of Gaussian random variables is the same as the sum of their individual expectations, which all are zero. A similar argument reveals that the variance is

$$E[b(t)^2] = \text{Var}\left[\sum_{i=1}^{n} x_i\right] = nT = \frac{t}{T}T = t. \tag{4.15}$$

The key step in (4.15) is recognizing that the number of steps n is equal to the time t divided by the time step T. Thus, we see that the variance incrases linearly as well move away from the $t = 0$. The Wiener process is often described as a random walk, and this behavior is illustrated in Figure 4.2.

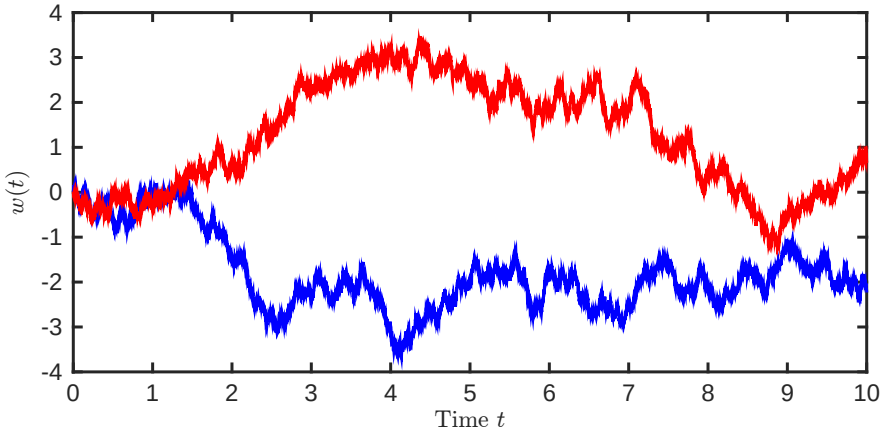


Figure 4.2: Two realizations of the Wiener process

■

■ **Example 4.3 — White Gaussian noise.** It is easy to define the concept of a *discrete-time* white Gaussian noise process, i.e., a white sequence of random variables: We just need to insist that the sequence consists of i.i.d. Gaussian random variables. For a *continuous-time process* $n(t)$ the concept of whiteness is more troublesome. First, we need a tangible definition. Here we define white noise as the derivative of the Wiener process:

$$n(t) = \lim_{\Delta \to 0} \frac{b(t+\Delta) - b(t)}{\Delta}. \tag{4.16}$$

We shall later show that this process indeed has the whiteness property when $\Delta \to 0$: In that limit, $n(t_1)$ is inependendent of $n(t_2)$ whenever $t_1 \neq t_2$. The reader may then ask: Why did we not simply define white Gaussian noise as the increments x_i that we used to construct the Wiener process? The answer has to do with the scaling of the variance. Let us take a look at the following two questions, which seem innocent enough:

a) What is $\mathrm{Var}\left[\int_0^s n(t)\mathrm{d}t\right]$?
b) What is $\mathrm{Var}\left[n(t)\right]$?

Question **a)** is important because we generally want to use white noise as a driving mechanism: In order to see how a linear system governed by a differential equation responds to white noise we must be able to say something meaningful about integrals such as the one in question **a)**. Clearly, in most applications, we would like this variance to be a finite number, neither zero nor infinity. Together with the requirement that $n(t_1)$ should be independent of $n(t_2)$ for all $t_1 \neq t_2$, this implies that the answer to question **b)** is infinity. To see this, let us first define the following approximation to the integral:

$$S = \sum_{k=0}^{t/\Delta} \Delta n(k\Delta) \approx \int_0^t n(\tau)\mathrm{d}\tau \tag{4.17}$$

The variance of S is then given by

$$\mathrm{Var}\left[S\right] = \mathrm{Var}\left[\sum_{k=0}^{t/\Delta} \Delta n(k\Delta)\right] = \Delta^2 \mathrm{Var}\left[\sum_{k=0}^{t/\Delta} n(k\Delta)\right] = \Delta^2 \left(\frac{t}{\Delta}\right) \mathrm{Var}\left[n(t)\right] \tag{4.18}$$

Solving for $\mathrm{Var}[n(t)]$ yields

$$\mathrm{Var}[n(t)] = \frac{1}{t\Delta}\mathrm{Var}[S] = \frac{1}{\Delta} \tag{4.19}$$

where the last step follows from the requirement that S should be identical with the Wiener process. For $n(t)$ to be truly white we need to take the limit $\Delta \to 0$, which leads to the blow-up of $\mathrm{Var}[n(t)]$. Therefore it would have been problematic to define white noise simply as the increments of the Wiener process. ∎

∎ **Example 4.4 — A random constant.** Let the function $f(t)$ be given by $f(t) = a$ where $a \sim \mathcal{N}(0,1)$. In contrast to the previous examples, this is a very simply stochastic process which does not require any sophisticated machinery. Again, it is interesting to study time integrals of the process. We find that

$$\mathrm{Var}\left[\int_0^\tau f(\tau)\mathrm{d}\tau\right] = \mathrm{Var}[at] = t^2. \tag{4.20}$$

By comparing this with (4.15) we see that the uncertainty caused by a integration of a fixed bias increases faster than the uncertainty caused by integration of white noise. This should not come as a surprise, since white noise has a self-cancelling property that the fixed bias does not have. ∎

4.3.1 Stationarity and the autocorrelation function

Let us then return to the general theory of stochastic properties. In principle, the pdf of an arbitrary stochastic process would need to involve an infinite number of entries. It would not only need to contain the full distributions of $\mathbf{x}(t)$ for all $t \in \mathbb{R}$, but it would also need to contain the joint distributions for all tuples $\mathbf{x}(t_1), \ldots, \mathbf{x}(t_k)$ for any number k. However, such a complicated specification is obviously not needed for the examples given above. We can restrict the general collection of arbitrary stochastic processes to classes of nice and tractable stochastic processes. One such class which obviously is nice to work with is the class of all Gaussian processes. Another, and unrelated, concept that tells us something about which processes are nicer than other processes, is stationarity. The literature makes a distinction between strict-sense-stationary and wide-sense-stationary. We will only study the latter kind of stationarity here, since it is most useful from a practical perspective.

Wide-sense stationary is described in terms of the autocorrelation function (ACF). For an arbitrary stochastic process it is defined as the expectation

$$R(t_1,t_2) = E[\mathbf{x}(t_1)\mathbf{x}(t_2)^\mathsf{T}]. \tag{4.21}$$

A stochastic process is said to be wide sense stationary if its mean is constant

$$E[\mathbf{x}(t)] = \eta \tag{4.22}$$

and the ACF can be written as a function of $\tau = t_2 - t_1$:

$$R(\tau) = E[\mathbf{x}(t)\mathbf{x}(t+\tau)^\mathsf{T}]. \tag{4.23}$$

In other words, for a wide sense stationary process, the correlations between $\mathbf{x}(t)$ at different times should only depend on the difference between these times. More explicitly, if we define $\mathbf{x}_1 = \mathbf{x}(t)$ and $\mathbf{x}_2 = \mathbf{x}(t+\tau)$, then we can also write the autocorrelation function as

$$R(\tau) = \int\int \mathbf{x}_1\mathbf{x}_2^\mathsf{T} p(\mathbf{x}_1,\mathbf{x}_2)\mathrm{d}\mathbf{x}_1\mathrm{d}\mathbf{x}_2. \tag{4.24}$$

Another nice property for a stochastic process to possess is Gaussianity. For a wide-sense stationary process this means that the joint density $p(\mathbf{x}_1, \mathbf{x}_2)$ always is a Gaussian. If the constant mean η is zero, then this distribution is of the form

$$p(\mathbf{x}_1, \mathbf{x}_2) = \mathcal{N}\left(\begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{bmatrix}; \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} R(0) & R(\tau) \\ R(\tau) & R(0) \end{bmatrix} \right). \tag{4.25}$$

Let us first look at the ACF of a non-stationary process.

Theorem 4.3.1 The Wiener process is non-stationary, and its ACF is given by $R(t_1, t_2) = \min(t_1, t_2)$.

Proof. Assume that $t_1 < t_2$. Since the Wiener process has independent increments we have that $E[b(t_1)(b(t_2) - b(t_1))] = 0$. On the other hand, we have that $E[b(t_1)(b(t_2) - b(t_1))] = E[b(t_1)b(t_2)] - E[b(t_1)^2] = E[b(t_1)b(t_2)] - t$. The last equality follows from (4.15). By comparing these two results it follows that $E[b(t_1)b(t_2)] = t$. The proof for the case when $t_1 > t_2$ is as good as identical. ■

Let us then show that white Gaussian noise is wide-sense stationary, and find its covariance.

Theorem 4.3.2 The ACF of the white noise process defined by (4.15) is given by $R(\tau) = \delta(\tau)$.

Proof. Let us first keep Δ fixed and define

$$\tilde{n}(t) = \frac{b(t+\Delta) - b(t)}{\Delta}. \tag{4.26}$$

The ACF of $\tilde{n}$ is then given by

$$\begin{aligned} E\left[\tilde{n}(t)\tilde{n}(t+\tau)\right] &= E\left[\frac{b(t+\Delta) - b(t)}{\Delta} \cdot \frac{b(t+\tau+\Delta) - b(t+\tau)}{\Delta} \right] \\ &= \frac{1}{\Delta^2} E\Big[b(t+\Delta)b(t+\tau+\Delta) - b(t)b(t+\tau+\Delta) \\ &\quad - b(t+\Delta)b(t+\tau) + b(t)b(t+\tau) \Big] \\ &= \frac{1}{\Delta^2} \Big(\min(t+\Delta, t+\Delta+\tau) - \min(t, t+\Delta+\tau) \\ &\quad - \min(t+\Delta, t+\tau) + \min(t, t+\tau) \Big) \end{aligned} \tag{4.27}$$

To find exact expressions for $E\left[\tilde{n}(t)\tilde{n}(t+d)\right]$ we have to distinguish between four cases:

1) $0 < \Delta < \tau$
2) $0 < \tau < \Delta$
3) $-\Delta < \tau < 0$
4) $\tau < -\Delta < 0$.

For the first case we find that

$$E\left[\tilde{n}(t)\tilde{n}(t+d)\right] = \frac{1}{\Delta}\Big(t + \Delta - t - (t+\Delta) + t \Big) = 0. \tag{4.28}$$

A similar argument yields 0 also for the fourth case. The middle cases require a bit more work. In the second case, the ACF is given by

$$E\left[\tilde{n}(t)\tilde{n}(t+\tau)\right] = \frac{1}{\Delta^2}\Big(t + \Delta - t - (t+\tau) + t \Big) = \frac{\Delta - \tau}{\Delta^2}. \tag{4.29}$$

In the third case, the ACF is given by

$$E\left[\tilde{n}(t)\tilde{n}(t+\tau)\right] = \frac{1}{\Delta^2}\left(t+\Delta+\tau-(t+\Delta+\tau)-(t+\tau)+t+\tau\right) = \frac{\Delta+\tau}{\Delta^2}. \tag{4.30}$$

We can sum this up as

$$E\left[\tilde{n}(t)\tilde{n}(t+\tau)\right] = \begin{cases} 0 & \text{if } 0<\Delta<\tau \\ \frac{1}{\Delta}\left(1-\frac{\tau}{\Delta}\right) & \text{if } 0<\tau<\Delta \\ \frac{1}{\Delta}\left(1+\frac{\tau}{\Delta}\right) & \text{if } -\Delta<\tau<0 \\ 0 & \text{if } \tau<-\Delta<0. \end{cases} \tag{4.31}$$

This is a "triangle pulse" which is entirely given by τ (and not by t). Thus, we have actually shown that white noise is stationary. To show that this triangle pulse indeed converges to the δ-function in the limit $\Delta \to 0$, we must verify that the integral of $E\left[\tilde{n}(t)\tilde{n}(t+\tau)\right]$ over τ is equal to one, no matter what Δ is. This is indeed the case:

$$\int_{-\Delta}^{0} \frac{1}{\Delta}\left(1+\frac{\tau}{\Delta}\right)\mathrm{d}\tau + \int_{0}^{\Delta} \frac{1}{\Delta}\left(1-\frac{\tau}{\Delta}\right)\mathrm{d}\tau = \frac{1}{2}+\frac{1}{2} = 1. \tag{4.32}$$

■

We can of course scale the strength of white noise: A white noise process with ACF $q\delta(\tau)$ is equal to a standard white noise process multiplied with $\sqrt{q}$. We refer to q as the variance of the white noise process, and $\sqrt{q}$ as the corresponding standard deviation.

4.3.2 Linear systems and the power spectral density

When a stochastic process is used as input to a dynamic system, the output will be another stochastic process. Recall from conventional (i.e., deterministic) state space theory that the response of a linear dynamical system can be modeled both in the time-domain and in the transform-domain. In the time domain the response to an arbitrary input signal is given by a convolution with the impulse response. In the transform domain the response is given by multiplication with the transfer function. The same rules applies when the input is a stochastic prosess. However, in this case we are mainly interested in how this affects the ACF or its Fourier transform, the power spectral density (PSD).

The convolution formulas for the ACF are given in the following theorem [70]. Notice that the theorem suggests evaluating the cross-correlation between the input and the output first, and then the autocorrelation of the output.

Theorem 4.3.3 Let the stochastic process $\mathbf{x}(t)$ with ACF $R_{xx}(t_1,t_2)$ be input to a linear system with impulse response $h(t)$ such that $\mathbf{y}(t) = \int_{-\infty}^{\infty} h(t-\tau)\mathbf{x}(\tau)\mathrm{d}\tau$. Assume that both $\mathbf{x}(t)$ and $h(t)$ are scalar and real-valued. The ACF $R_{yy}(t_1,t_2)$ of $\mathbf{y}(t)$ is then given by

$$R_{xy}(t_1,t_2) = \int_{-\infty}^{\infty} R_{xx}(t_1,t_2-\alpha)h(\alpha)\mathrm{d}\alpha \tag{4.33}$$

$$R_{yy}(t_1,t_2) = \int_{-\infty}^{\infty} R_{xy}(t_1-\alpha,t_2)h(\alpha)\mathrm{d}\alpha. \tag{4.34}$$

Proof. We only prove (4.33). From commutativity of the convolution operation and linearity of the integral it follows that

$$\mathbf{x}(t_1)\mathbf{y}(t_2) = \int \mathbf{x}(t_1)h(\alpha)\mathbf{x}(t_2-\alpha)\mathrm{d}\alpha.$$

Taking the expectation yields

$$E[\mathbf{x}(t_1)\mathbf{y}(t_2)] = \int E[\mathbf{x}(t_1)h(\alpha)\mathbf{x}(t_2-\alpha)]\mathrm{d}\alpha = \int h(\alpha)R_{xx}(t_1,t_2-\alpha)\mathrm{d}\alpha.$$

Proving (4.34) follows along very similar steps. ■

■ **Example 4.5 — The Gauss-Markov process.** As an example of this theorem we can look at the case when white noise $v(t)$ with ACF $R_{vv}(\tau) = q\delta(\tau)$ is used as input to a linear system with impulse response $h(t) = e^{-ct}u(t)$ where $u(t)$ is the Heaviside function. Since no real system has infinite memory, we assume that the white noise is switched on at $t = 0$. Therefore, we define $\mathbf{x}(t) = v(t)u(t)$ as the actual input to the system. The ACF of $\mathbf{x}(t)$ is given by

$$R_{xx}(t_1,t_2) = \begin{cases} 0 & \text{if } t_1 < 0 \text{ or } t_2 < 0 \\ R_{vv}(t_2-t_1) & \text{otherwise.} \end{cases} = R_{vv}(t_2-t_1)u(t_1)u(t_2) \tag{4.35}$$

Notice that $v(t)$ is a stationary stochastic process while $\mathbf{x}(t)$ is a non-stationary stochastic process. Defining $\mathbf{y}(t)$ as the convolution of $h(t)$ and $\mathbf{x}(t)$, our next step is to evaluate the ACF $R_{xy}(t_1,t_2)$. It is given by

$$\begin{aligned} R_{xy}(t_1,t_2) &= \int_{-\infty}^{\infty} R_{xx}(t_1,t_2-\alpha)h(\alpha)\mathrm{d}\alpha \\ &= \int_{-\infty}^{\infty} R_{vv}(t_2-\alpha-t_1)u(t_1)u(t_2-\alpha)h(\alpha)\mathrm{d}\alpha \\ &= \int_{-\infty}^{\infty} q\delta(t_2-t_1-\alpha)u(t_1)u(t_2-\alpha)e^{-c\alpha}u(\alpha)\mathrm{d}\alpha \\ &= qe^{-c(t_2-t_1)}u(t_1)u(t_2-t_1) \end{aligned} \tag{4.36}$$

The second equality in (4.36) follows from inserting (4.35) in the formula for R_{xy}, and accordingly replace t_2 with $t_2-\alpha$. In the third equality we have inserted concrete expressions for R_{vv} and h. Notice that a third Heaviside $u(\alpha)$ enters the picture because it is part of the impulse response. To arrive at the fourth equality we use the sifting property of the delta function and notice that the middle Heaviside then becomes redundant. Notice that $R_{xy}(t_1,t_2)$ according to (4.36) is zero whenever whenever $t_1 < 0$, and also whenever $t_2 < t_1$. This reflects the fact that $\mathbf{y}(t)$ only contains contributions from $\mathbf{x}(t)$ at previous time steps. In the remainder of the example, we therefore assume that $0 < t_1 < t_2$. We proceed to insert R_{xy} into the ACF for the filter output.

$$\begin{aligned} R_{yy}(t_1,t_2) &= \int_{-\infty}^{\infty} qe^{-c\alpha}e^{-c(t_2-t_1+\alpha)}u(\alpha)u(t_1-\alpha)u(t_2-t_1+\alpha)\mathrm{d}\alpha \\ &= qe^{-c(t_2-t_1)}\int_{-\infty}^{\infty} e^{-2c\alpha}u(\alpha)u(t_1-\alpha)u(t_2-t_1+\alpha)\mathrm{d}\alpha. \end{aligned} \tag{4.37}$$

The triple product of Heaviside functions need careful treatment. The first Heaviside yields a lower integration limit of zero. The second Heaviside can be rewritten as $u(t_1-\alpha) = 1-u(\alpha-t_1)$. Before dealing with the third Heaviside we recall the assumption that $t_1 < t_2$. This Heaviside can then be rewritten as $u(t_2-t_1+\alpha) = u(\alpha-(t_1-t_2))$ where $t_1-t_2 < 0$. Thus, this Heaviside becomes irrelevant since its changepoint is lower than for the first Heaviside. This leaves us with

$$R_{yy}(t_1,t_2) = qe^{-c(t_2-t_1)}\int_0^{t_1} e^{-2c\alpha}\mathrm{d}\alpha = \frac{q}{2c}(1-e^{-ct_1})e^{-c(t_2-t_1)} \tag{4.38}$$

which is our final expression for the ACF of the Gauss-Markov process. Recall that this expression is valid if $0 < t_1 < t_2$. Otherwise, the ACF is zero. It is interesting to notice that the term e^{-ct_1} goes towards zero as t_1 increases. In the limit as $t_1 \to \infty$ the Gauss-Markov process indistinguishable from a stationary process with ACF $\frac{q}{2c}e^{-c(t_2-t_1)}$. ■

Convolutions are in general cumbersome to evaluate. Furthermore, the expressions in Theorem 4.3.3 are only applicable to scalar systems. As in deterministic system theory, we can overcome these challenges by working in the transform domain. The main role is then played by the PSD, which is defined as the Fourier transform of the ACF:

$$S(\omega) = \int_{-\infty}^{\infty} R(\tau) e^{-\mathrm{i}\omega\tau} \mathrm{d}\tau \quad \Leftrightarrow \quad R(\tau) = \frac{1}{2\pi} \int_{-\infty}^{\infty} S(\omega) e^{\mathrm{i}\omega\tau} \mathrm{d}\omega. \tag{4.39}$$

The name power spectral density describes the fact that the PSD tells us how the energy of a signal is distributed among different frequencies (this is known as the Wiener-Khinchin theorem). Notice that $S(\omega)$ is only defined for wide-sense stationary processes, because the definition takes for granted that the ACF can be written as

$$R(t_1, t_2) = R(t_2 - t_1) = R(\tau) \text{ where } \tau = t_2 - t_1. \tag{4.40}$$

As in deterministic linear systems theory, it is also useful to define the transfer function, which is the Fourier transform of the impulse response:

$$H(\omega) = \int_{-\infty}^{\infty} h(\tau) e^{-\mathrm{i}\omega\tau} \mathrm{d}\tau. \tag{4.41}$$

To derive formulas similar to Theorem 4.3.3 in the frequency domain, let us first rewrite the expressions for $R_{xy}(t_1, t_2)$ and $R_{yy}(t_1, t_2)$ in the wide-sense stationary case:

$$R_{xy}(\tau) = \int R_{xx}(\tau - \alpha) h(\alpha) \mathrm{d}\alpha \tag{4.42}$$

$$R_{yy}(\tau) = \int R_{xy}(\tau - \alpha) h(-\alpha) \mathrm{d}\alpha. \tag{4.43}$$

The verify these expression is left as a small exercise to the reader.

Theorem 4.3.4 Let the wide-sense stationary stochastic process $\mathbf{x}(t)$ be input to a linear system with impulse response $h(t)$ such that $\mathbf{y}(t) = \int_{-\infty}^{\infty} h(t-\tau)\mathbf{x}(\tau)\mathrm{d}\tau$. Assume that both $\mathbf{x}(t)$ and $h(t)$ are scalar and real-valued. Let $S_{xx}(\omega)$ be the PSD of $\mathbf{x}(t)$, and let $H(\omega)$ be the Fourier transform of $h(t)$. Then $\mathbf{y}(t)$ is also wide-sense stationary and its PSD is given by

$$S_{xy}(\omega) = H(\omega) S_{xx}(\omega) \quad \text{and} \quad S_{yy}(\omega) = H^*(\omega) S_{xy}(\omega). \tag{4.44}$$

Proof. This follows directly from taking the Fourier transform of (4.42) and (4.43) by means of the convolution theorem, and utilizing the fact that if $H(\omega)$ is the Fourier transform of $h(t)$, then $H^*(\omega)$ is the Fourier transform of $h(-t)$. ■

In other words, we can find the PSD according to

$$S_{yy}(\omega) = H(\omega) S_{xx}(\omega) H^*(\omega) = |H(\omega)|^2 S_{xx}(\omega). \tag{4.45}$$

■ **Example 4.6 — PSD of white noise.** We recall that the ACF of white noise is a δ-function $q\delta(\tau)$. The PSD is the Fourier transform of this, which is a constant

$$S_{ww}(\omega) = q. \tag{4.46}$$

The physical interpretation of this is that white noise has energy equally distributed among all frequencies, from zero to infinity. Clearly this is not possible in the real world, and it demonstrates that white noise is a mathematical abstraction that must be treated with some care. ■

■ **Example 4.7 — PSD of Gauss-Markov process.** We recall that the Gauss-Markov process was generated by passing white noise with strength q through a linear filter with impulse response $h(t) = e^{ct}u(t)$. We know from linear systems theory that the corresponding transfer function is

$$H(\omega) = \frac{1}{c + \mathrm{i}\omega} \;\Rightarrow\; |H(\omega)]^2 = \frac{1}{c^2 + \omega^2}. \tag{4.47}$$

To find the ACF, we should multiply this with q and take the inverse transform. To do this, we may consult any table of Fourier transform pairs, which hopefully contains the pair

$$e^{-c|\tau|} \;\leftrightarrow\; \frac{2c}{c^2 + \omega^2}. \tag{4.48}$$

Notice that the function $e^{-c|t|}$ is different from $h(t)$ because it also is nonzero for negative τ. Thus we arrive at the ACF

$$R_{yy}(\tau) = \frac{q}{2c} e^{-c|\tau|}. \tag{4.49}$$

This is identical to the ACF that we obtained by means of convolution in Example 4.5 when we omit the middle factor $(1 - e^{-ct_1})$. The reason for the discrepancy is that we in Example 4.5 replaced the stationary pure white noise input with a non-stationary version defined by $\mathbf{x}(t) = v(t)u(t)$ in order to maintain causality. As $t_1 \to \infty$ we see that the middle term tends towards unity, and the two ACFs become indistinguishable. This means that in the long run we can treat the Gauss-Markov process as a stationary process. This would not be possible for, e.g., the Wiener process. ■

From comparing Examples 4.5 and 4.7 we see that obtaining the ACF of a process that comes out of a linear filter is much easier in the frequency domain than in the time domain. The price to pay is that we must confine ourselves to stationary processes.

There are at least two other reasons why the frequency domain representation of stochastic processes can be useful. First, we may want to estimate the correlation properties of a given time series of noise samples. By working in the frequency domain we get access to many tools of system identification. Particular frequencies (from, e.g., a pattern of ocean waves) that characterize the data are much easier to spot in the empirical PSD than in the empirical ACF. Second, we may want to simulate noise with given correlation properties. In general it takes less computational resources to do this in the frequency domain than in the time domain. We do not delve any deeper into these topics here, as this is material that more naturally would belong in a course in digital signal processing.

Related to simulation is also modeling. If we know that our process noise has a certain PSD, then we can find a transfer function $H(\omega)$ such that (4.45) holds, and we can then construct a differential equation that would have $H(\omega)$ or some approximation of it as its transfer function. For example, for the Gauss-Markov process the obvious differential equation would be

$$\dot{y} = -cy + w, \;\; w \sim \mathcal{N}(0, q\delta(t - \tau)) \tag{4.50}$$

where the notation $\delta(t - \tau)$ signifies that w is a white noise process.

4.4 Continuous-time models

In typical filtering problems the process model (4.7) is naturally specified in continuous time using the framework of stochastic processes from the previous section. A continuous-time linear process model is of the form

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u} + \mathbf{G}\mathbf{n} \tag{4.51}$$

where $\mathbf{x}$ is the state vector, $\mathbf{u}$ is the control input and $\mathbf{n}$ is the process noise. To specify such a model we need to specify the elements in $\mathbf{x}$, the matrices $\mathbf{A}$, $\mathbf{B}$ (often zero) and $\mathbf{G}$, and the stochastic properties of $\mathbf{n}$. The process noise is typically assumed to be white, as is the case in Example 4.8. If the noise indeed is correlated, we extend the state vector with the correlated noise process, and include a filter that would produce noise with the desired correlation properties as part of the state space model, see Example 4.9.

■ **Example 4.8 — The CV model.** In the tracking literature, the most common kinematic model is the 2-dimensional constant velocity (CV) model. In continuous time it is given by $\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{G}\mathbf{n}$ where $\mathbf{x}$ is the state vector and $\mathbf{n}$ is the process noise. The matrices are given by

$$\mathbf{A} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad \text{and} \quad \mathbf{G} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{4.52}$$

while the process noise is assumed white with diagonal covariance, i.e.,

$$\mathbf{n} \sim \mathcal{N}(\mathbf{0}, \mathbf{D}\delta(t-\tau)) \quad \text{where} \quad \mathbf{D} = \begin{bmatrix} \sigma_a^2 & 0 \\ 0 & \sigma_a^2 \end{bmatrix}. \tag{4.53}$$

The number σ_a is a measure of how much acceleration the target is expected to undergo. Thus, knowledge about the likely acceleration can be used as a crude way of determining σ_a. Notice that this model essentially integrates white noise. The last two states in $\mathbf{x}$ are therefore independent Wiener processes. ■

■ **Example 4.9 — Accelerometer with slowly varying bias.** In this example, we look at inertial navigation of a vehicle that moves along a straight road. Let the state vector consist of the following states:

$$\mathbf{x} = \begin{bmatrix} \text{Position of the vehicle} \\ \text{Velocity of the vehicle} \\ \text{Bias of the accelerometer} \end{bmatrix}$$

The accelerometer provides measurements of acceleration. Since these measurements are the derivatives of velocity, they cannot be modeled as a linear combination of elements in the state vector. Instead, the accelerometer measurements are modeled as a control input. The corresponding noise on the accelerometer measurements must then enter the system as process noise, and not as *bona-fide* measurement noise. Matters are complicated by the fact that accelerometers tend to suffer from slowly varying biases. An obvious candidate for modeling this is the Gauss-Markov process from example 4.5. All of this leads to a continuous-time model of the form $\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u} + \mathbf{G}\mathbf{n}$ where

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & -c \end{bmatrix}, \ \mathbf{B} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \ \mathbf{G} = \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{4.54}$$

and where

$$\mathbf{n} \sim \mathcal{N}(\mathbf{0}, \mathbf{D}\delta(t-\tau)) \text{ where } \mathbf{D} = \begin{bmatrix} \sigma_a^2 & 0 \\ 0 & \sigma_b^2 \end{bmatrix}. \tag{4.55}$$

Here $\mathbf{u}$ are the accelerometer readings extrapolated as a continuous-time signal and $\mathbf{n}$ is the plant noise. The model is governed by the constants c, σ_a and σ_b. If we define $T = \frac{1}{c}$ as the time constant of the bias process, and we know that the mean square value of the bias is b, then we find that

$$\sigma_b^2 = 2bc = \frac{2b}{T}. \tag{4.56}$$

The standard deviation σ_a quantifies the uncertainty that remains in the accelerometer readings when the bias is accounted for. In the above model, this uncertainty is modeled as white Gaussian noise. ∎

4.5 Discretization

Continuous-time models such as those from the last two examples are useful from a conceptual perspective because they are invariant to the discretization time interval. Nevertheless, discretization is required for practical implementation and state estimation by means of the Kalman filter as formulated in Section 4.2. More precisely, discretization is the task of deriving the Markov model $p(\mathbf{x}_k \,|\, \mathbf{x}_{k-1})$ from the continuous time model (4.51), where $\mathbf{x}_k = \mathbf{x}(t_k)$ and $\mathbf{x}_{k-1} = \mathbf{x}(t_{k-1})$. For notational simplicity, less us also denote the discretization interval by

$$T = t_k - t_{k-1}. \tag{4.57}$$

From basic control theory, it is evident that the solution of (4.51) can be written as

$$\mathbf{x}_k = \mathbf{F}\mathbf{x}_{k-1} + \mathbf{u}_k + \mathbf{v}_k \tag{4.58}$$

where

$$\mathbf{F} = e^{\mathbf{A}(t_k - t_{k-1})}, \;\; \mathbf{u}_k = \int_{t_{k-1}}^{t_k} e^{\mathbf{A}(t_k-\tau)}\mathbf{B}\mathbf{u}(\tau)\,\mathrm{d}\tau \;\; \text{and} \;\; \mathbf{v}_k = \int_{t_{k-1}}^{t_k} e^{\mathbf{A}(t_k-\tau)}\mathbf{G}\mathbf{n}(\tau)\,\mathrm{d}\tau. \tag{4.59}$$

Evaluation of $\mathbf{F}$ and $\mathbf{u}_k$ is straightforward. All the randomness, conditional on $\mathbf{x}_{k-1}$, is encapsulated in $\mathbf{v}_k$. To complete the specification of $p(\mathbf{x}_k \,|\, \mathbf{x}_{k-1})$ we therefore have to quantify the statistical properties of $\mathbf{v}_k$. It is fairly obvious that the discrete-time plant noise inherits whiteness and Gaussianity from its continuous-time counterpart. All we need to know about $\mathbf{v}_k$ is then encapsulated in its instantaneous covariance matrix $\mathbf{Q}$.

Theorem 4.5.1 — Discretization of LTI stochastic systems. Let the continuous-time model be given by (4.51) where $\mathbf{n} \sim \mathcal{N}(\mathbf{0}, \mathbf{D}\delta(t-\tau))$. Also, assume that the discretization time is fixed at $T = t_k - t_{k-1}$. The covariance matrix of $\mathbf{v}_k$ in the discrete-time model (4.58) is then given by

$$\mathbf{Q} = E[\mathbf{v}_k \mathbf{v}_k^\mathsf{T}] = \int_0^T e^{(T-\tau)\mathbf{A}}\mathbf{G}\mathbf{D}\mathbf{G}^\mathsf{T} e^{(T-\tau)\mathbf{A}^\mathsf{T}}\,\mathrm{d}\tau \tag{4.60}$$

Proof. We prove the result in the slightly more general context of establishing the correlation between the discrete-time plant noise contributions at two different time steps. Following along the lines of page 189 in [3], we have that

$$\begin{aligned}
E[\mathbf{v}_k \mathbf{v}_l^\mathsf{T}] &= E\left[\left(\int_{t_{k-1}}^{t_k} e^{\mathbf{A}(t_k-\tau_1)}\mathbf{G}\mathbf{n}(\tau_1)\,\mathrm{d}\tau_1\right)\left(\int_{t_{l-1}}^{t_l} e^{\mathbf{A}(t_l-\tau_2)}\mathbf{G}\mathbf{n}(\tau_2)\,\mathrm{d}\tau_2\right)^\mathsf{T}\right] \\
&= E\left[\int_{t_{k-1}}^{t_k}\int_{t_{l-1}}^{t_l} e^{\mathbf{A}(t_k-\tau_1)}\mathbf{G}\mathbf{n}(\tau_1)\mathbf{n}(\tau_2)^\mathsf{T}\mathbf{G}^\mathsf{T} e^{\mathbf{A}^\mathsf{T}(t_l-\tau_2)}\mathrm{d}\tau_1\mathrm{d}\tau_2\right] \\
&= \int_{t_{k-1}}^{t_k}\int_{t_{l-1}}^{t_l} e^{\mathbf{A}(t_k-\tau_1)}\mathbf{G}E\left[\mathbf{n}(\tau_1)\mathbf{n}(\tau_2)^\mathsf{T}\right]\mathbf{G}^\mathsf{T} e^{\mathbf{A}^\mathsf{T}(t_l-\tau_2)}\mathrm{d}\tau_1\mathrm{d}\tau_2 \\
&= \int_{t_{k-1}}^{t_k}\int_{t_{l-1}}^{t_l} e^{\mathbf{A}(t_k-\tau_1)}\mathbf{G}\mathbf{D}\delta(\tau_1-\tau_2)\mathbf{G}^\mathsf{T} e^{\mathbf{A}^\mathsf{T}(t_l-\tau_2)}\mathrm{d}\tau_1\mathrm{d}\tau_2 \\
&= \int_{t_{k-1}}^{t_k} e^{(t_k-\tau)\mathbf{A}}\mathbf{G}\mathbf{D}\mathbf{G}^\mathsf{T} e^{(t_k-\tau)\mathbf{A}^\mathsf{T}}\mathrm{d}\tau\,\delta_{kl}.
\end{aligned} \tag{4.61}$$

Shifting the integration limits and recognizing the Kronecker delta leads to formula in the theorem. Notice that the third line of (4.61) is valid even if $\mathbf{n}$ is colored, while whiteness is used to make the transition to the fourth line. ■

It is not immediately obvious how one should evaluate the integral in Theorem 4.5.1. In a first-order approximation we could assume that $e^{(t_k-\tau)\mathbf{A}} \approx \mathbf{I}$, which would lead to

$$\mathbf{Q} \approx \mathbf{G}\mathbf{D}\mathbf{G}^{\mathsf{T}} T. \tag{4.62}$$

This is in general not recommended. The matrix product $\mathbf{G}\mathbf{G}^{\mathsf{T}}$ is not necessarily positive definite, even if the true $\mathbf{Q}$ in (4.61) is positive definite. However, a closed-form solution is also possible, and is generally to be preferred over approximations whenever one is in possession efficient techniques for evaluating the matrix exponential:

Theorem 4.5.2 — Van Loan's formula. Let us define the matrices $\mathbf{V}_1$ and $\mathbf{V}_2$ according to

$$\exp\left(\begin{bmatrix} -\mathbf{A} & \mathbf{G}\mathbf{D}\mathbf{G}^{\mathsf{T}} \\ \mathbf{0} & \mathbf{A}^{\mathsf{T}} \end{bmatrix} T\right) = \begin{bmatrix} \times & \mathbf{V}_2 \\ \mathbf{0} & \mathbf{V}_1 \end{bmatrix}. \tag{4.63}$$

Then we can find $\mathbf{Q}$ from Theorem 4.5.1 according to $\mathbf{Q} = \mathbf{V}_1^{\mathsf{T}}\mathbf{V}_2$.

Proof. A proof of the theorem can be found in Section II of Van Loan's article [88]. The proof is nontrivial, and does not provide much insights into genuine sensor fusion issues, and is therefore not repeated here. ■

The presence of a matrix exponential in this formula means that everything is not yet entirely resolved. Different programming languages have different levels of support for evaluating matrix exponentials. In some cases, the direct evaluation of a matrix exponential may be to slow to be of real-time utility. Sometimes it is possible to evaluate (4.63) exactly in terms of elementary functions. For example, for the discrete-time CV model all terms of order 4 and above in the series expansion s of the matrix exponential in (4.63) become zero, so that it can be truncated to order 3. For this model, the transition matrix and discrete-time plant noise covariance become

$$\mathbf{F} = \begin{bmatrix} 1 & 0 & T & 0 \\ 0 & 1 & 0 & T \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \text{and} \quad \mathbf{Q} = \begin{bmatrix} \frac{T^3}{3} & 0 & \frac{T^2}{2} & 0 \\ 0 & \frac{T^3}{3} & 0 & \frac{T^2}{2} \\ \frac{T^2}{2} & 0 & T & 0 \\ 0 & \frac{T^2}{2} & 0 & T \end{bmatrix} \sigma_a^2. \tag{4.64}$$

The derivation of these matrices is given to the reader as Exercise 4.2.

4.6 Tuning of the Q matrix

Having established the fundamental procedure for converting a continuous-time plant model into a discrete-time process model, we also have to provide sensible values for process noise parameters such as σ_a^2 in (4.64). This is made tricky by the fact that white noise ultimately is a dubious description of reality. For example, the acceleration of a moving object will be more of an on-off kind of thing than a pure white noise process. However, any such acceleration pattern is a possible realization from the ensemble of a white noise process. Let us once again keep in mind that white noise with variance q really is a limit of Wiener process increments as the time step goes towards zero. Thinking in terms of such increments, it is clear that a realization containing many increments larger than the standard deviation will have lower probability than a realization where

the increments are similar to or smaller than the standard deviation. From this mechanism we could expect that a large value of q would be more plausible to explain whatever increments there are.

On the other hand, assuming a smaller value of q, which is tantamount to assuming that the increments on average are smaller, would concentrate more probability mass at realizations that involve small increments. Thus, a balance results, between the plausibility of a large q to explain increments observed, and the plausibility of a small q in the absence of increments. A rigorous approach to determining this equilibrium can be found in maximum likelihood estimation, which we will study in Section 4.8.

In practical applications such as target tracking it is, however, of limited value to have a theoretically optimal estimate of the plant noise strength according to such considerations, because the white noise construction after all is so artificial. The actual process noise will typically vary a lot with time. While methods exist to deal with this variability, see Chapter 6, it is desirable at least from the perspective of simplicity to have a fixed plant noise covariance. Setting the plant noise covariance low will yield better accuracy when no significant maneuvers find place. Setting it high will yield inferior accuracy in this situation, but may be necessary to achieve acceptable performance during stronger maneuvers. A simple guideline that can be used is therefore to set the plant noise as high as required to account for the strongest maneuvers that are likely to happen.

■ **Example 4.10 — The maneuverability of a human diver.** The following example is taken from [16], where an active sonar was used to track a human diver with re-breather system. the diver is capable of going from resting to swimming in about one second, and is able to sustain a speed of approximately $0.5\,\mathrm{m/s}$. We see this happening in Figure 4.3. A CV model was used to track the diver.

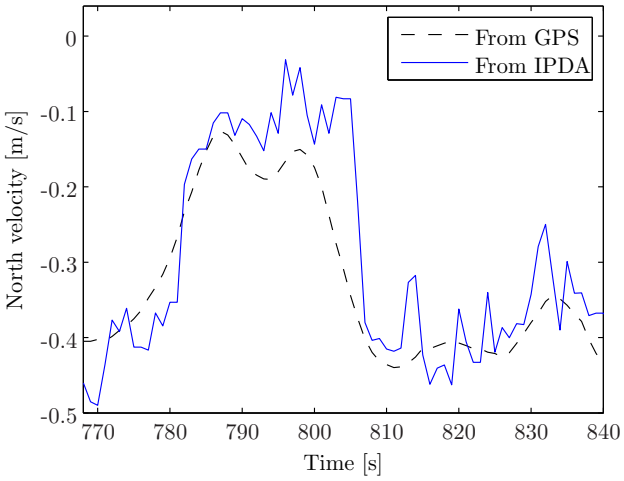


Figure 4.3: Target velocity for the diver scenario over a limited time interval.

Based on these considerations, $\sigma_a = 0.5\,\mathrm{m/s^2}$ appears to be reasonable value for the process noise, which was again used in the tracker that generated the results in Figure 4.3. The tracking method is described in greater detail in Section 7.5. ■

4.6.1 Filter consistency

While simple physical considerations as in Example 4.10 are always mandatory and often sufficient, a more systematic way of tuning the $\mathbf{Q}$ matrix is by means of *filter consistency*. We say that a filter is consistent if its errors on average are well described by the output of the filter. This leads to the following requirements:

1. The state errors should be acceptable as zero mean.

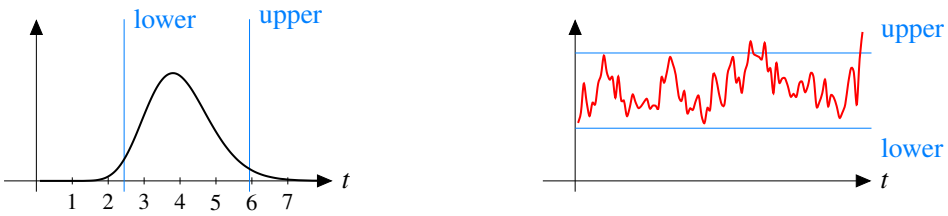


Figure 4.4: On the left: χ^2 distributions with confidence intervals. On the right: The average of ε_k over 10 Monte-Carlo simulations compared with the corresponding χ^2-boundaries.

2. The state errors should have magnitude commensurate with the state covariance yielded by the filter.
3. The innovations should be acceptable as zero mean.
4. The innovations should have magnitude commensurate with the innovation covariance yielded by the filter.
5. The innovations should be acceptable as white.

The typical test for consistency is to check whether criteria 2 or 4 holds. We refer the reader to [3] pp. 235-236 for bias and whiteness tests. The standard tools for testing criteria 2 and 4 are the normalized estimation error squared (NEES)

$$\varepsilon_k = (\hat{\mathbf{x}}_k - \mathbf{x}_k)^{\mathsf{T}} \mathbf{P}_k^{-1} (\hat{\mathbf{x}}_k - \mathbf{x}_k) \tag{4.65}$$

and the normalized innovations squared (NIS)

$$\varepsilon_k^{\nu} = \boldsymbol{\nu}_k^{\mathsf{T}} \mathbf{S}_k^{-1} \boldsymbol{\nu}_k. \tag{4.66}$$

In the remainder of this section, suppose that $\mathbf{x} \in \mathbb{R}^d$ and that $\mathbf{z} \in \mathbb{R}^s$. If the filter model indeed is correct, then ε_k and ε_k^{ν} will be χ^2 distributed random variables with d and s, respectively, as their degrees of freedoms (cf., Exercise 3.4). Furthermore, when we add together several ε_k or ε_k^{ν}, either from N different time steps or from N different realizations, then we get a new χ^2 distributed random variable with Nd or Ns degrees of freedom (cf., Exercise 2.2). The *average* NEES or NIS (known as ANEES and ANIS) should therefore obey a scaled χ^2 distribution, or more precisely a Gamma distribution with scale parameter $2/N$ and shape parameter $Nd/2$ or $Ns/2$. For any such distribution we can construct a confidence interval between the α quantile and the $1-\alpha$ quantile for a sensible value of α. Say we are looking for a 95% confidence interval for the ANEES. Then its lower and upper bound can be found according to

$$\texttt{lower} = \texttt{chi2inv}(0.025, Nd)/N \quad \text{and} \quad \texttt{upper} = \texttt{chi2inv}(0.975, Nd)/N \tag{4.67}$$

using the Matlab function for inverse χ^2 cdf. Assuming that everything else, including the measurement noise covariance, is correct, we can expect too large values in **Q** to push the ANEES below `lower`, while too small values in **Q** to push the ANEES above `upper`. Both situations are undesirable as they will lead to suboptimal estimation, but the latter is the most serious situation. If the ANEES tends to be too high it means that $\mathbf{P}_k$ tends to be too low, meaning that the Kalman filter is overconfident. Both in target tracking (with its measurement origin uncertainty) and in inertial navigation (with its nonlinearities) this is a common cause of divergence. In any case, a consistency analysis should always be carried out when designing a Kalman filter, as it also can be a useful debugging technique.

■ **Example 4.11 — Tuning of the CV model in the Autosea tracker.** In [82] this procedure was used to determine the plant noise strength for a CV model used in a maritime radar tracking system.

In this work, a Kalman filter based on CV model was used for measurements from the automatic identification system (AIS). The AIS measurements included both position and velocity, leading to the measurement matrix $\mathbf{H} = \mathbf{I}_4$. To compensate for quantization effects[1], the measurement noise covariance was set to

$$\mathbf{R} = \begin{bmatrix} 0.5^2 & 0 & 0 & 0 \\ 0 & 0.5^2 & 0 & 0 \\ 0 & 0 & 0.1^2 & 0 \\ 0 & 0 & 0 & 0.1^2 \end{bmatrix} + \frac{1}{12} \begin{bmatrix} v_x^2 & 0 & 0 & 0 \\ 0 & v_y^2 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}. \tag{4.68}$$

Kalman filters were then used to track the following ships with their AIS data as measurements:

Name	Type	Length × Breadth	Mean SOG (m/s)
GLUTRA	Passenger	94.8 m × 16.0 m	5.3
SULA	Cargo	87.9 m × 12.8 m	5.7
KORSFJORD	Passenger	122 m × 16.7 m	5.6
TRONDHEIMSFJORD II	High speed	24.5 m × 8 m	12.8
TELEMETRON	Pleasure craft	8 m × 3 m	13.1

Figure 4.5: Ships used to determine the plant noise in the Autosea tracker.

The Kalman filters were given both $\sigma_a = 0.05$ and $\sigma_a = 0.5$ as plant noise strength. Based on the tracking results, average NIS values and corresponding normalized χ^2 intervals (r_1, r_2) were calculated for the two values of σ_a:

Name	$\sigma_a = 0.05$		$\sigma_a = 0.5$		(r_1, r_2)	N
	NIS	AI	NIS	AI		
GLUTRA	**4.67**	−0.02	0.90	0.01	(3.47, 4.55)	109
SULA	**3.61**	−0.22	0.51	−0.10	(3.49, 4.56)	106
KORSFJORD	71.8	−1.33	**4.31**	−0.44	(3.52, 4.51)	127
TR.FJORD II	11.3	−0.62	**3.24**	−0.16	(3.76, 4.24)	533
TELEMETRON	371	−0.04	**4.45**	−0.01	(3.77, 4.23)	579

Figure 4.6: Consistency results

For the slower ships GLUTRA and SULA, the low value of $\sigma_a = 0.05$ appears most adequate. However, this value led to unacceptable NIS-values for the ships KORSFJORD, TRONDHEIMSFJORD II and TELEMETRON. When $\sigma_a = 0.5$ is used, the average NIS resides inside its 95% confidence interval for KORSFJORD, while it is a little bit too low for TRONDHEIMSFJORD II and a little bit too high for TELEMETRON. This value of σ_a was therefore chosen. The price to be paid for this is suboptimal tracking performance for slower ships such as GLUTRA and SULA: Their tracks will fluctuate more than they would have done if the lower value of σ_a had been chosen. ∎

A final remark is in order with regard to filter consistency. Any kind of model mismatches, not only a bad $\mathbf{Q}$ matrix, can lead to bad consistency properties. For nonlinear filters, the underlying approximation techniques will also have an impact on consistency. This concept is therefore very important in most of the subsequent chapters.

[1] AIS messages are given time stamps measured in an integer number of seconds, see [82] for details.

4.7 Tuning of the R matrix

The measurement noise is generally more tangible than the process noise. Any sensor has finite resolution, which yields a lower bound for the elements in the measurement noise matrix $\mathbf{R}$. The resolution is typically quantized (e.g., resolution cells for a radar), so that the measurements in principle will be discrete-valued random variables. To maintain the assumption of Gaussian measurement noise, this discrete measurement model must be approximated by a Gaussian. This is typically done by moment matching.

■ **Example 4.12** Consider a scenario with point targets and a 2-dimensional sensor that covers the surveillance region with square cells of fixed resolution Δx. Assume that the state vector $\mathbf{x}$ contains positions and velocities according to $\mathbf{x} = [x, y, v_x, v_y]^\mathsf{T}$ With the measurement matrix $\mathbf{H} = [\mathbf{I}_2, \mathbf{0}]$ we find that the distribution of $\mathbf{z} - \mathbf{H}\mathbf{x}$ conditional on $\mathbf{x}$ is uniform according to

$$p(\mathbf{z} - \mathbf{H}\mathbf{x}|\mathbf{x}) = \begin{cases} 1/\Delta x^2 & \text{if } \|\mathbf{z} - \mathbf{H}\mathbf{x}\|_\infty < \Delta x/2 \\ 0 & \text{otherwise.} \end{cases} \tag{4.69}$$

This distribution can be approximated by a Gaussian with the same covariance, with is

$$\mathbf{R} = \begin{bmatrix} \frac{\Delta x^2}{12} & 0 \\ 0 & \frac{\Delta x^2}{12} \end{bmatrix}. \tag{4.70}$$

With this approximation, (4.69) can be replaced by

$$p(\mathbf{z}|\mathbf{x}) = \mathcal{N}(\mathbf{z}; \mathbf{H}\mathbf{x}, \mathbf{R}) \tag{4.71}$$

where $\mathbf{z}$ is to be understood as the centroid of the corresponding resolution cell. ■

In reality, a sensor's resolution cells do not normally constitute squares or rectangles in Cartesian space. For example, the resolution cells of a 2-dimensional surveillance radar represent constant range and bearing. In other words, the resolution cells are rectangles in polar coordinates. This makes the filtering problem nonlinear. One possible solution is to use nonlinear filtering techniques such as the extended Kalman filter, which we will discuss in the next chapter. An alternative is to convert the measurements from polar coordinates to Cartesian coordinates, so that the standard Kalman filter still can be used. Several conversion methods have been proposed [2] [54] [58]. These methods perform both transformation of the measurement noise covariance, and also *bias compensation* due to the fact that an ellipse in polar coordinates becomes a banana in Cartesian coordinates. If the sensor resolution is sufficiently fine, there is no real need to conduct bias compensation, and we can transform the covariance matrix by pre- and post-multiplication by the Jacobian of polar-to-Cartesian mapping.

■ **Example 4.13** Assume that the measurement model is of the form $\mathbf{z}_k = \mathbf{h}(\mathbf{x}_k) + \mathbf{w}_k$ where $\mathbf{w}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{R})$, and where

$$\mathbf{h}(\mathbf{x}_k) = \begin{bmatrix} \sqrt{x^2 + y^2} \\ \operatorname{atan2}(y, x) \end{bmatrix} \quad \text{and} \quad \mathbf{R} = \begin{bmatrix} \sigma_r^2 & 0 \\ 0 & \sigma_\theta^2 \end{bmatrix}. \tag{4.72}$$

In other words, the measurements are given in polar coordinates, and we denote the elements in $\mathbf{z}_k$ correspondingly: $\mathbf{z}_k = [r, \theta]^\mathsf{T}$. Again, assume that the state vector is $\mathbf{x} = [x, y, v_x, v_y]^\mathsf{T}$. The converted measurement model, where the measurements are given in Cartesian coordinates without debiasing, is then of the form $\mathbf{y}_k = \mathbf{H}\mathbf{x}_k + \mathbf{w}_k$ where $\mathbf{w}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{R}_c)$, and where

$$\mathbf{y}_k = \begin{bmatrix} r\cos\theta \\ r\sin\theta \end{bmatrix}, \quad \mathbf{H} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \quad \text{and} \quad \mathbf{R}_c = \mathbf{J}\mathbf{R}\mathbf{J}^\mathsf{T}, \quad \text{where} \quad \mathbf{J} = \frac{\partial}{\partial \mathbf{z}_k}\mathbf{h}^{-1}(\mathbf{z}_k). \tag{4.73}$$

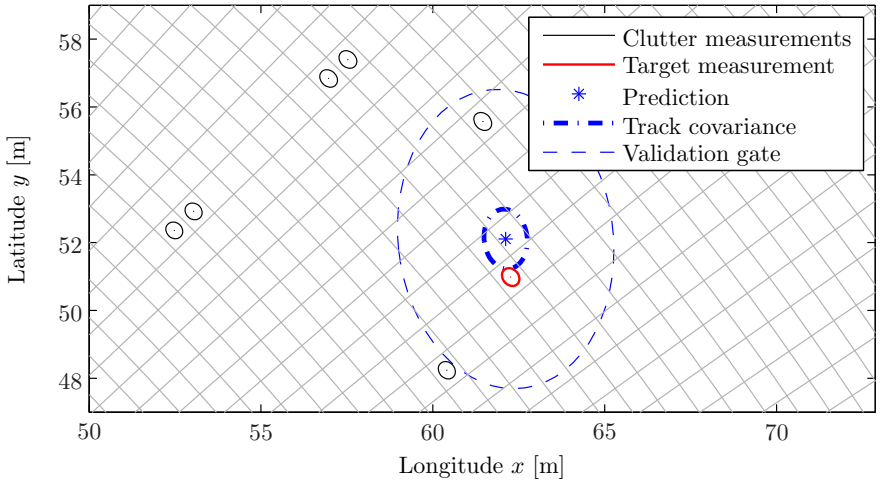


Figure 4.7: Illustration of Gaussian distributions matched to the discrete resolution cells in sonar tracking [14]. In this figure we also see a track prediction with corresponding uncertainty (blue stuff). We will study this in greater detail in Chapters 7 and 8.

By differentiating $\mathbf{h}^{-1}(\mathbf{z}_k)$, which simply is the mapping from $\mathbf{z}_k$ to $\mathbf{y}_k$, we find the the elements of $\mathbf{J}$ to be

$$\mathbf{J} = \begin{bmatrix} \cos\theta & -r\sin\theta \\ \sin\theta & r\cos\theta \end{bmatrix}. \tag{4.74}$$

A combined visualization of what is going on in Examples 4.12 and 4.13 is shown in Figure 4.7. ∎

To further complicate matters, sensor resolution may not be the only source of uncertainty that needs to be accounted for by $\mathbf{R}$. In target tracking and SLAM, a single target or landmark may cover several resolution cells. The standard approach then is to extract the centroids of local "blobs" as measurements by means of a clustering procedure. It is in general hard or impossible to provide analytical expressions for the resulting uncertainty, and one must resort to empirical investigations. This can be done means of real data similar to those the system is expected to handle, or simulations from a realistic sensor model that includes the entire measurement extraction machinery. For an example of the former, the reader is referred to [82]. For an example of the latter, the reader is referred to [16].

4.8 System identification and joint estimation of Q and R

Theorem 4.8.1 — Maximum likelihood estimation of system parameters. Consider the discrete-time state-space model (4.6), and assume that both $\mathbf{Q}$ and $\mathbf{R}$ depend on an unknown parameter vector $\boldsymbol{q}$. The maximum likelihood estimate of $\boldsymbol{q}$, if it exists, can then be found as

$$\boldsymbol{q}_{\mathrm{ML}} = \arg\max_{\boldsymbol{q}} \sum_k \log \mathcal{N}(\mathbf{z}_k; \mathbf{H}\hat{\mathbf{x}}_{k|k-1}, \mathbf{S}_k) \tag{4.75}$$

where $\hat{\mathbf{x}}_{k|k-1}$ and $\mathbf{S}_k$ are as given in Algorithm 1.

Proof. The likelihood of all the recieved measurements conditional on $\boldsymbol{q}$ is $p(\mathbf{z}_{1:k} \,|\, \boldsymbol{q})$. As a first step, we expand it as an integral over all the unknown states $\mathbf{x}_1, \ldots, \mathbf{x}_k$ in accordance with the total

probability theorem:

$$p(\mathbf{z}_{1:k} \,|\, \boldsymbol{q}) = \int p(\mathbf{z}_{1:k} \,|\, \mathbf{x}_{1:k}, \boldsymbol{q}) p(\mathbf{x}_{1:k}, \boldsymbol{q}) \,\mathrm{d}\mathbf{x}_{1:k} \tag{4.76}$$

By utilizing the Markov structure, we can write it in greater detail as

$$p(\mathbf{z}_{1:k} \,|\, \boldsymbol{q}) = \int \left(\prod_{l=1}^{k} p(\mathbf{z}_l \,|\, \mathbf{x}_l, \boldsymbol{q}) \right) \left(\prod_{l=2}^{k} p(\mathbf{x}_l \,|\, \mathbf{x}_{l-1}, \boldsymbol{q}) \right) p(\mathbf{x}_1) \,\mathrm{d}\mathbf{x}_{1:k}. \tag{4.77}$$

Furthermore, by invoking the Gaussian-linear assumptions of the model (4.6), it becomes

$$p(\mathbf{z}_{1:k} \,|\, \boldsymbol{q}) = \int \left(\prod_{l=1}^{k} \mathcal{N}(\mathbf{z}_l \,;\, \mathbf{H}\mathbf{x}_l, \mathbf{R}) \right) \left(\prod_{l=2}^{k} \mathcal{N}(\mathbf{x}_l \,;\, \mathbf{F}\mathbf{x}_{l-1}, \mathbf{Q}) \right) \mathcal{N}(\mathbf{x}_1 \,;\, \hat{\mathbf{x}}_{1|0}, \mathbf{P}_{1|0}) \,\mathrm{d}\mathbf{x}_{1:k}. \tag{4.78}$$

For simplicity, we have removed reference to the parameter vector $\boldsymbol{q}$. We can rewrite this expression as a sequence of nested integrals $I_1, I_2, \ldots$ so that

$$\begin{aligned}
I_1(\mathbf{x}_2) &= \int \mathcal{N}(\mathbf{x}_2 \,;\, \mathbf{F}\mathbf{x}_1, \mathbf{Q}) \mathcal{N}(\mathbf{z}_1 \,;\, \mathbf{H}\mathbf{x}_1, \mathbf{R}) \mathcal{N}(\mathbf{x}_1 \,;\, \hat{\mathbf{x}}_{1|0}, \mathbf{P}_{1|0}) \,\mathrm{d}\mathbf{x}_1, \\
I_2(\mathbf{x}_3) &= \int \mathcal{N}(\mathbf{x}_3 \,;\, \mathbf{F}\mathbf{x}_2, \mathbf{Q}) \mathcal{N}(\mathbf{z}_2 \,;\, \mathbf{H}\mathbf{x}_2, \mathbf{R}) I_1(\mathbf{x}_2) \,\mathrm{d}\mathbf{x}_2 \\
&\vdots \\
p(\mathbf{z}_{1:k} \,|\, \boldsymbol{q}) &= \int \mathcal{N}(\mathbf{z}_k \,;\, \mathbf{H}\mathbf{x}_k, \mathbf{R}) I_{k-1}(\mathbf{x}_k) \,\mathrm{d}\mathbf{x}_k.
\end{aligned} \tag{4.79}$$

By utilizing the product identity as used in Theorems 4.2.1 and 4.2.2 we can write the first integral as

$$I_1(\mathbf{x}_2) = \mathcal{N}(\mathbf{z}_1 \,;\, \mathbf{H}\hat{\mathbf{x}}_{1|0}, \mathbf{S}_1) \mathcal{N}(\mathbf{x}_2 \,;\, \mathbf{F}, \mathbf{F}\mathbf{P}_{1|0}\mathbf{F}^{\mathsf{T}} + \mathbf{Q}). \tag{4.80}$$

The second integral becomes

$$I_2(\mathbf{x}_3) = \mathcal{N}(\mathbf{z}_1 \,;\, \mathbf{H}\hat{\mathbf{x}}_{1|0}, \mathbf{S}_1) \mathcal{N}(\mathbf{z}_2 \,;\, \mathbf{H}\hat{\mathbf{x}}_{2|1}, \mathbf{S}_2) \mathcal{N}(\mathbf{x}_2 \,;\, \mathbf{F}, \mathbf{F}\mathbf{P}_{2|1}\mathbf{F}^{\mathsf{T}} + \mathbf{Q}). \tag{4.81}$$

and so on. We see that as we progress through the time indices we accumulate factors of the form $\mathcal{N}(\mathbf{z}_l \,;\, \mathbf{H}\hat{\mathbf{x}}_{l|l-1}, \mathbf{S}_l)$, and by taking the logarithm the product of these factors becomes the sum in the theorem. ■

▪ **Example 4.14** To study how maximum likelihood estimation of $\mathbf{Q}$ and $\mathbf{R}$ may work in practice, let us attempt to estimate the tuning parameters σ_z and σ_a in a CT model with straightforward Cartesian position measurements:

$$\dot{\mathbf{x}} = \begin{bmatrix} \mathbf{0}_{2\times 2} & \mathbf{I}_2 \\ \mathbf{0}_{2\times 2} & \mathbf{0}_{2\times 2} \end{bmatrix} \mathbf{x} + \begin{bmatrix} \mathbf{0}_{2\times 2} \\ \mathbf{I}_2 \end{bmatrix} \mathbf{n}, \;\; \mathbf{z}_k = \begin{bmatrix} \mathbf{I}_2 & \mathbf{0}_{2\times 2} \end{bmatrix} + \mathbf{w} \tag{4.82}$$

where

$$\mathbf{n} \sim \mathcal{N}(\mathbf{0}, \sigma_a \mathbf{I}_2 \delta(t-\tau)) \;\; \text{and} \;\; \mathbf{w}_k \sim \mathcal{N}(\mathbf{0}, \sigma_z \mathbf{I}_2 \delta_{kl}). \tag{4.83}$$

We use the discretization interval $T = 1$ and 40 time steps, and $\sigma_z = 5$ and $\sigma_a = 0.2$ as ground truth parameters. We then plot the likelihood, i.e., the exponential of the sum in (4.75), as a function of possible values of σ_a and σ_z. For two different random seeds the results shown in Figure 4.8 were

obtained. We see that the estimates of σ_z are fairly sharp: In both cases it seems unlikely that σ_z should deviate more than, say, 20% from the maximum likelihood estimate. The estimates on σ_a are, on the other hand, much less reliable. In the simulation used in the left hand side of Figure 4.8, it seems clear that σ_a must be relatively close to its ground truth. But in the simulation used on the right hand side, much lower values of σ_a appear to be more plausible.

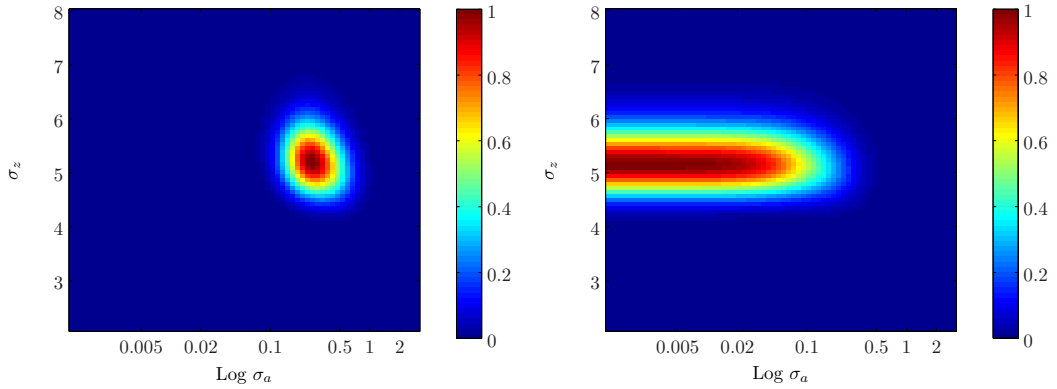


Figure 4.8: The likelihood function, normalized relative to its maximum, for two realizations of the CV model.

■

This example demonstrates that it may or may not be possible to obtain reliable estimates for the model parameters used in a Kalman filter by means of maximum likelihood. It is therefore generally preferable to use physical domain knowledge and common sense to set such values, as done in Sections 4.5 and 4.7. If there nevertheless is a need or desire to estimate these values from data, it may be a good idea to analyze whether the parameters actually can be estimated with a reasonable accuracy. A possible tool for this analysis is the Cramer Rao Lower Bound (CRLB) [73].

4.9 LTI and LTV systems

We have tacitly assumed that the matrices $\mathbf{A}$, $\mathbf{F}$, $\mathbf{Q}$, $\mathbf{R}$ and so on are constant in time. This means that we are looking at linear time-invariant (LTI) systems. Such systems are of course the easiest systems to analyze. In particular, most of the tools from frequency domain analysis require the systems to be of this form. However, the Kalman filter does not depend on this restriction. The Kalman filter remains an optimal estimator also for linear time-variant (LTV) systems where the matrices are allowed to change in time, insofar as they are known. In the discrete-time case we should then use a notation such as $\mathbf{A}_k$, $\mathbf{F}_k$, $\mathbf{Q}_k$, $\mathbf{R}_k$ and so on to emphasize the variability with time. Many sources (e.g., [76]) do indeed use such a notation when they introduce the Kalman filter in order to strive for utmost generality. In this book the philosophy is a bit different: We avoid "cluttered notation" as much as possible, and take for granted that the reader will be able to understand the most obvious generalizations.

In contrast to the LTI and the LTV cases, now consider the case where there is uncertainty in the system matrices. Also consider the case where the system matrices depend on the state vector. In these cases the system is neither LTI nor LTV, but nonlinear. Then the Kalman filter is no longer optimal, and possible not even usable at all, and we may want to consider nonlinear filtering techniques such as those described in the next chapter.

4.10 References and chapter remarks

The treatment of the main topic of this chapter, the Kalman filter, is largely based on Chapter 5 in [3]. The product identity is not used to derive the Kalman filter in [3]. Instead, it is derived as a case of MMSE estimation for Gaussian random variables. While the mathematical details are very similar to our derivation of the product identity, there is a huge conceptual difference: Our derivation of the Kalman filter is purely based on probability, and does not rely on any concepts related to estimators and estimation. Another perspective on the Kalman filter can be found in [39], where the Kalman filter is derived from the criterion that it should minimize the trace of the covariance of the state estimate. Finally, we could also have arrived at the Kalman filter from the orthogonality principle that we used to derive the LMMSE estimator.

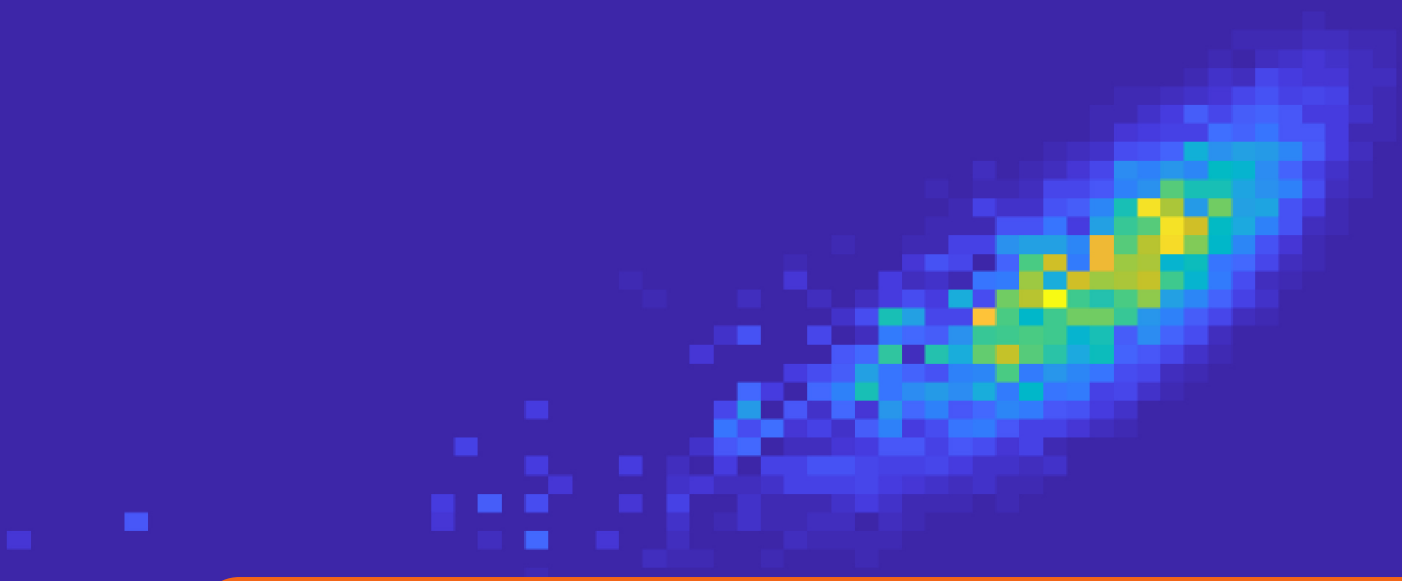
The treatment of stochastic processes is largely taken from [70]. For wide-sense stationary processes, one must decide on a sign convention for the cross-correlation $R_{xy}(\tau)$. While [70] uses the convention $\tau = t_1 - t_2$ we have used the convention $\tau = t_2 - t_1$, which also is used by Wikipedia.

The use of filter consistency as a tuning tool is recommended in [3]. If one instead wants to go for a maximum likelihood technique then [79] could be a reference worth checking out.

4.11 Exercises

Exercise 4.1 In inertial navigation, an accelerometer suffers from a slowly varying bias according to the Gauss-Markov model (4.54). What should the driving noise be to model a bias with average strength $1\,\mathrm{m/s^2}$, if the time constant is $T = 1\,\mathrm{s}$? ∎

Exercise 4.2 Derive the transition matrix $\mathbf{F}$ and discrete-time covariance $\mathbf{Q}$ for the CV model that are given in (4.64). ∎



5. Non-linear filtering

In the real world, estimation problems are seldom simple enough to fit the Gaussian-linear model studied in Chapter 4. Depending on how serious the deviations from Gaussianity and linearity are, a variety of filtering techniques are available. The purpose of this chapter is to review non-linear filtering techniques that are commonly used in target tracking.

First of all it should be mentioned that even if a problem technically is nonlinear, none of these techniques may be required. For example, if measurements are given in polar coordinates, a simple coordinate conversion such as the one that was suggested in Section 4.7 is most often preferable to nonlinear filtering techniques.

The main focus in this chapter is on two nonlinear filtering techniques: the Extended Kalman filter (EKF) and particle filters. The former is commonly used for problems whose nonlinearities are easy to linearize. The latter is a very general methodology that can be used for virtually any Bayesian filtering problem which involves a Markov model and a likelihood. Both have their own weaknesses, which may or may not lead to unacceptable performance degradation. Both of the two fundamental approaches, i.e., linearization and sampling, can be combined and expanded, leading to a wealth of different nonlinear estimation methods.

5.1 Linearization and the Extended Kalman Filter

It is often possible to attack nonlinearities by means of linearization, so that linear estimation techniques still can be used. Different options are possible for the choice of linearization technique. In some cases it can make sense to linearize around a fixed operating point. Such a fixed linearization point could be the initial estimate of a constant parameter, or the origin for a system that always remains so close to the origin that this linearization remains reasonably valid. In contrast, the operating point that the EKF linearizes around is the most recent estimate for the prediction step, and the most recent prediction for the update step.

Before we delve into the details of the EKF, let us also realize that more complex or sophisticated options for linearization exist as well. Having linearized once around the most recent prediction, and then performed an EKF update, we can once again linearize around the new estimate, and calculate the measurement update around this linearization point, and we can proceed in this manner

until we are sufficiently happy. Finally, we can also obtain a linear model by means of the LMMSE philosophy from Section 2.6.5. The we do not linearize around an operating point at all. Instead, we build the linear model by means of moment approximations, which typically are found from deterministic sample-based integration techniques.

The EKF is typically used for systems whose Markov model and measurement model can be written as follows:

$$\begin{aligned} \mathbf{x}_k &= \boldsymbol{f}(\mathbf{x}_{k-1}) + \mathbf{v}_k, & \mathbf{v}_k &\sim \mathcal{N}(0, \mathbf{Q}) \\ \mathbf{z}_k &= \mathbf{h}(\mathbf{x}_k) + \mathbf{w}_k, & \mathbf{w}_k &\sim \mathcal{N}(0, \mathbf{R}). \end{aligned} \tag{5.1}$$

Comparing with the model (4.9) from the previous chapter, we see that $\boldsymbol{f}$ and $\mathbf{h}$ are nonlinear functions, as opposed to the linear matrices $\mathbf{F}$ and $\mathbf{H}$ in (4.9). In order to construct linearized approximations of $\boldsymbol{f}$ and $\mathbf{h}$, we first introduce the error variables

$$\Delta\mathbf{x}_{k-1} = \mathbf{x}_{k-1} - \hat{\mathbf{x}}_{k-1} \tag{5.2}$$

$$\Delta\mathbf{x}_k = \mathbf{x}_k - \hat{\mathbf{x}}_{k|k-1} \tag{5.3}$$

and then make Taylor expansions in terms of the error variables:

$$\boldsymbol{f}(\mathbf{x}_{k-1}) \approx \boldsymbol{f}(\hat{\mathbf{x}}_{k-1}) + \mathbf{F}(\hat{\mathbf{x}}_{k-1})\,\Delta\mathbf{x}_{k-1} \tag{5.4}$$

$$\mathbf{h}(\mathbf{x}_k) \approx \mathbf{h}(\hat{\mathbf{x}}_{k|k-1}) + \mathbf{H}(\hat{\mathbf{x}}_{k|k-1})\,\Delta\mathbf{x}_k. \tag{5.5}$$

The matrices $\mathbf{F}$ and $\mathbf{H}$ are now found as Jacobians of $\boldsymbol{f}$ and $\mathbf{h}$ with respect to $\mathbf{x}_{k-1}$ and $\mathbf{x}_k$:

$$\mathbf{F}(\hat{\mathbf{x}}_{k-1}) = \left.\frac{\partial}{\partial \mathbf{x}_{k-1}} \boldsymbol{f}(\mathbf{x}_{k-1})\right|_{\mathbf{x}_{k-1}=\hat{\mathbf{x}}_{k-1}} \tag{5.6}$$

$$\mathbf{H}(\hat{\mathbf{x}}_{k|k-1}) = \left.\frac{\partial}{\partial \mathbf{x}_k} \mathbf{h}(\mathbf{x}_k)\right|_{\mathbf{x}_k=\hat{\mathbf{x}}_{k|k-1}} \tag{5.7}$$

In the sequel we omit the dependencies of the Jacobians on $\hat{\mathbf{x}}_{k-1}$ and $\hat{\mathbf{x}}_{k|k-1}$ for notational simplicity. The prediced density can then be expressed as

$$\begin{aligned} p(\mathbf{x}_k|\mathbf{z}_{1:k-1}) &\approx \int \mathcal{N}(\mathbf{x}_k - \boldsymbol{f}(\hat{\mathbf{x}}_{k-1})\,;\, \mathbf{F}\Delta\mathbf{x}_{k-1}, \mathbf{Q})\mathcal{N}(\Delta\mathbf{x}_{k-1}\,;\, \mathbf{0}, \mathbf{P}_{k-1})\mathrm{d}\Delta\mathbf{x}_{k-1} \\ &= \mathcal{N}(\mathbf{x}_k\,;\, \boldsymbol{f}(\hat{\mathbf{x}}_{k-1}), \mathbf{F}\mathbf{Q}\mathbf{F}^\mathsf{T} + \mathbf{Q}) \int \mathcal{N}(\Delta\mathbf{x}_{k-1}\,;\, \ldots)\mathrm{d}\Delta\mathbf{x}_{k-1} \\ &= \mathcal{N}(\mathbf{x}_k\,;\, \boldsymbol{f}(\hat{\mathbf{x}}_{k-1}), \mathbf{F}\mathbf{Q}\mathbf{F}^\mathsf{T} + \mathbf{Q}) \end{aligned} \tag{5.8}$$

while the posterior density can be expressed as

$$\begin{aligned} p(\mathbf{x}_k|\mathbf{z}_{1:k}) &\propto \mathcal{N}(\mathbf{z}_k\,;\, \mathbf{h}(\mathbf{x}_k), \mathbf{R})\mathcal{N}(\mathbf{x}_k\,;\, \hat{\mathbf{x}}_{k|k-1}, \mathbf{P}_{k|k-1}) \\ &\approx \mathcal{N}(\mathbf{z}_k - \mathbf{h}(\hat{\mathbf{x}}_{k|k-1})\,;\, \mathbf{H}\Delta\mathbf{x}_k, \mathbf{R})\mathcal{N}(\Delta\mathbf{x}_k\,;\, \mathbf{0}, \mathbf{P}_{k|k-1}) \\ &= \mathcal{N}(\ldots)\mathcal{N}(\Delta\mathbf{x}_k\,;\, \mathbf{0} + \mathbf{W}(\mathbf{z}_k - \mathbf{h}(\hat{\mathbf{x}}_{k|k-1}) - \mathbf{H}\mathbf{0}), (\mathbf{I} - \mathbf{W}\mathbf{H}\mathbf{P}_{k|k-1}) \\ &\propto \mathcal{N}(\mathbf{x}_k\,;\, \hat{\mathbf{x}}_{k|k-1} + \mathbf{W}(\mathbf{z}_k - \mathbf{h}(\hat{\mathbf{x}}_{k|k-1})), (\mathbf{I} - \mathbf{W}\mathbf{H}\mathbf{P}_{k|k-1}) \end{aligned} \tag{5.9}$$

where the Kalman gain is given by the usual expression $\mathbf{W}_k = \mathbf{P}_{k|k-1}\mathbf{H}^\mathsf{T}\mathbf{S}_k^{-1}$. Again, the product identity has been used to obtain the desired results. In the same way as we did for the standard KF, we also express the EKF in algorithmic form below:

Algorithm 2 The extended Kalman filter

procedure EKF($\hat{\mathbf{x}}_{k-1}$, $\mathbf{P}_{k-1}$, $\mathbf{z}_k$)
  $\hat{\mathbf{x}}_{k|k-1} \leftarrow \boldsymbol{f}(\hat{\mathbf{x}}_{k-1})$ ▷ The predicted state estimate
  $\mathbf{F} \leftarrow \frac{\partial}{\partial \mathbf{x}_{k-1}} \boldsymbol{f}(\mathbf{x}_{k-1})$ ▷ The prediction Jacobian
  $\mathbf{P}_{k|k-1} \leftarrow \mathbf{F}\mathbf{P}_{k-1}\mathbf{F}^\mathsf{T} + \mathbf{Q}$ ▷ The predicted covariance
  $\hat{\mathbf{z}}_{k|k-1} \leftarrow \mathbf{h}(\hat{\mathbf{x}}_{k|k-1})$ ▷ The predicted measurement
  $\boldsymbol{\nu}_k \leftarrow \mathbf{z}_k - \hat{\mathbf{z}}_{k|k-1}$ ▷ The innovation
  $\mathbf{H} \leftarrow \frac{\partial}{\partial \mathbf{x}_k} \mathbf{h}(\mathbf{x}_k)$ ▷ The measurement Jacobian
  $\mathbf{S}_k \leftarrow \mathbf{H}\mathbf{P}_{k|k-1}\mathbf{H}^\mathsf{T} + \mathbf{R}$ ▷ The innovation covariance
  $\mathbf{W}_k \leftarrow \mathbf{P}_{k|k-1}\mathbf{H}^\mathsf{T}\mathbf{S}_k^{-1}$ ▷ The Kalman gain
  $\hat{\mathbf{x}}_k \leftarrow \hat{\mathbf{x}}_{k|k-1} + \mathbf{W}_k\boldsymbol{\nu}_k$ ▷ The posterior state estimate
  $\mathbf{P}_k \leftarrow (\mathbf{I} - \mathbf{W}_k\mathbf{H})\mathbf{P}_{k|k-1}$ ▷ The posterior covariance
  return $\hat{\mathbf{x}}_k$, $\mathbf{P}_k$, $\hat{\mathbf{x}}_{k|k-1}$, $\mathbf{P}_{k|k-1}$, $\mathbf{S}_k$
end procedure

■ **Example 5.1 — EKF for the CT model.** In this example we shall introduce the coordinated turn (CT) model, which in target tracking is the most important alternative to the CV model. The fundamental premise behind this model is that the target moves with a nearly constant turn-rate, which we denote ω. We use the right-hand convention that a positive turn rate results when the cross product of two consecutive velocity vectors has a positive z-component. The following relationships between velocities and accelerations can then be derived:

$$\ddot{x} = -\omega\dot{y} \quad \text{and} \quad \ddot{y} = \omega\dot{x}. \tag{5.10}$$

If we define a state vector as $\mathbf{y} = [x, y, \dot{x}, \dot{y}]^\mathsf{T}$ then (5.10) can be rewritten as a state space model of the form

$$\dot{\mathbf{y}} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & -\omega \\ 0 & 0 & \omega & 0 \end{bmatrix} \mathbf{y}. \tag{5.11}$$

So far, this is a purely linear system. If we include ω in the state vector, then it becomes a nonlinear system. Also, (5.11) is a purely deterministic system. We make it into a a stochastic system by introducing plant noise processes for x- and y-accelerations with standard deviation σ_a, and also for the turn rate ω with standard deviation σ_ω. Furthermore, we assume that the turn rate evolves according to a Wiener process multiplied by σ_ω. With the state vector $\mathbf{x} = [x, y, \dot{x}, \dot{y}, \omega]^\mathsf{T}$, this leads to the nonlinear plant model

$$\dot{\mathbf{x}} = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -\omega & 0 \\ 0 & 0 & \omega & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \mathbf{x} + \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \mathbf{n} \tag{5.12}$$

where $\mathbf{n} \sim \mathcal{N}(\mathbf{0}, \mathbf{D})$ and $\mathbf{D} = \operatorname{diag}([\sigma_a^2, \sigma_a^2, \sigma_\omega^2])$. The model is nonlinear because the turn rate ω both appears in the state vector and in the system matrix. An exact discretization of (5.12) is therefore not possible. However, if we assume that ω is slowly varying, in other words that σ_ω is sufficiently small compared to the time step T, then it should be a fair approximation to decouple

the dynamics of x, y, $\dot{x}$ and $\dot{y}$ from the dynamics of ω, leading to two linear models. Stitching these together, and calculating matrix exponentials, we arrive at

$$\mathbf{x}_k = \begin{bmatrix} 1 & 0 & \frac{\sin T\omega_{k-1}}{\omega_{k-1}} & \frac{-1+\cos T\omega_{k-1}}{\omega_{k-1}} & 0 \\ 0 & 1 & \frac{1-\cos T\omega_{k-1}}{\omega_{k-1}} & \frac{\sin T\omega_{k-1}}{\omega_{k-1}} & 0 \\ 0 & 0 & \cos T\omega_{k-1} & -\sin T\omega_{k-1} & 0 \\ 0 & 0 & \sin T\omega_{k-1} & \cos T\omega_{k-1} & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \mathbf{x}_{k-1} + \mathbf{v}_k \tag{5.13}$$

where the discrete-time plant noise $\mathbf{v}_k$ is zero-mean Gaussian white with covariance matrix

$$\mathbf{Q} = \begin{bmatrix} \frac{T^3}{3}\sigma_a^2 & 0 & \frac{T^2}{2}\sigma_a^2 & 0 & 0 \\ 0 & \frac{T^3}{3}\sigma_a^2 & 0 & \frac{T^2}{2}\sigma_a^2 & 0 \\ \frac{T^2}{2}\sigma_a^2 & 0 & T\sigma_a^2 & 0 & 0 \\ 0 & \frac{T^2}{2}\sigma_a^2 & 0 & T\sigma_a^2 & 0 \\ 0 & 0 & 0 & 0 & T\sigma_\omega^2 \end{bmatrix} \tag{5.14}$$

Again, (5.13) is a nonlinear model due to the presence of ω_{k-1} both in the transition matrix and in the previous state vector $\mathbf{x}_{k-1}$. An annoying artifact of (5.13) the fact that it is not defined for $\omega_{k-1} = 0$. Nevertheless, the limit value can be calculated using L'Hopital's rule, or we may simple take the matrix exponential of (5.12) with $\omega = 0$, and (5.13) then boils down to the CV model.

To implement an EKF for (5.13) we have to calculate the Jacobian for the mapping from $\mathbf{x}_{k-1}$ to $\mathbf{x}_k$. While somewhat tedious, the calculations are in principle straightforward[1], and lead to

$$\mathbf{F} = \begin{bmatrix} 1 & 0 & \frac{1}{\omega}\sin T\omega & -\frac{1}{\omega}(1-\cos T\omega) & \mathbf{F}_{1,5} \\ 0 & 1 & \frac{1}{\omega}(1-\cos T\omega) & \frac{1}{\omega}\sin T\omega & \mathbf{F}_{2,5} \\ 0 & 0 & \cos T\omega & -\sin T\omega & \mathbf{F}_{3,5} \\ 0 & 0 & \sin T\omega & \cos T\omega & \mathbf{F}_{4,5} \\ 0 & 0 & 0 & 0 & \mathbf{F}_{5,5} \end{bmatrix} \tag{5.15}$$

where the last column in $\mathbf{F}$ is

$$\mathbf{F}_{\cdot,5} = \begin{bmatrix} \mathbf{F}_{1,5} \\ \mathbf{F}_{2,5} \\ \mathbf{F}_{3,5} \\ \mathbf{F}_{4,5} \\ \mathbf{F}_{5,5} \end{bmatrix} = \begin{bmatrix} \frac{1}{\omega^2}(\dot{y} - (T\omega\dot{y} + \dot{x})\sin T\omega + (T\omega\dot{x} - \dot{y})\cos T\omega) \\ \frac{1}{\omega^2}(-\dot{x} + (\omega T\dot{x} - \dot{y})\sin T\omega + (\omega T\dot{y} + \dot{x})\cos T\omega) \\ -T(\dot{x}\sin T\omega + \dot{y}\cos T\omega) \\ T(\dot{x}\cos T\omega - \dot{y}\sin T\omega) \\ 1 \end{bmatrix}. \tag{5.16}$$

For notational simplicity the time index $k-1$ has been dropped in (5.15) and (5.16). Again, special attention is required to deal with the limit $\omega \to 0$. After wrestling a bit with the maths, the following matrix results:

$$\lim_{\omega\to 0} \mathbf{F} = \begin{bmatrix} 1 & 0 & T & 0 & -\frac{T^2\dot{y}}{2} \\ 0 & 1 & 0 & T & \frac{T^2\dot{x}}{2} \\ 0 & 0 & 1 & 0 & -T\dot{y} \\ 0 & 0 & 0 & 1 & T\dot{x} \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.17}$$

[1] I recommend that you always verify Jacobian calculations both using a symbolic mathematics program such as Maple, and using numerical differentiation in, e.g., Matlab.

Table 5.1: Parameter values used in CT EKF example

Symbol	Description	Value
σ_a	Acceleration plant noise standard deviation	$0.02\ \mathrm{m/s^2}$
σ_ω	Turn rate plant noise standard deviation	$0.005\ /\ \mathrm{s}$
σ_z	Measurement noise standard deviation	5 m
T	Discretization time interval	0.5 s

Notice that the upper left 4×4 sub-matrix in (5.17) is the transition matrix from the CV model. We have now specified all the mathematics that is geniune to the CT model. To design an EKF for the CT model we must also specify a measurement model. We opt for standard position measurements, i.e.,

$$\mathbf{z}_k = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{bmatrix} \mathbf{x}_k + \mathbf{w}_k, \ \ \mathbf{w}_k \sim \mathcal{N}(\mathbf{0}, \sigma_z^2 \mathbf{I}_2). \tag{5.18}$$

We benchmark the CT model against the CV model using simulations. We investigate a simulated scenario with the model parameters given in able 5.1. The CV model uses the same values for σ_a, σ_z and T as the CT model. We draw the initial state vector according to $\mathcal{N}(\hat{\mathbf{x}}_1, \mathbf{P}_1)$ where $\hat{\mathbf{x}}_1 = [0, 0, 5, 0, 0.05]^\mathsf{T}$ and $\mathbf{P}_1 = \mathrm{diag}([25, 25, 0.25, 0.25, 0.0025])$. In Figure 5.1 we see the comparison of the CT model with the CV model in a single Monte-Carlo simulation. For fairness of comparison, the CT filter model is initialized with zero turn rate, i.e., with $\hat{\mathbf{x}}_1 = [0, 0, 5, 0, 0]^\mathsf{T}$. We see that the CT model nevertheless has a fairly good concept of the actual turn rate after only three measurements. The CV model is also able to follow the turn, but it lags with $5 - 10$ m, which may or may not be acceptable. Notice the different sizes of the covariance ellipses in Figure 5.1. The CV model gives a smaller covariance than the CT model despite being worse off. In other words, the CV model has poor consistency properties. Keep in mind that the CV model was implemented with the same acceleration process noise as the CT model. Increasing the *assumed* process noise for the CV model would improve its consistency.

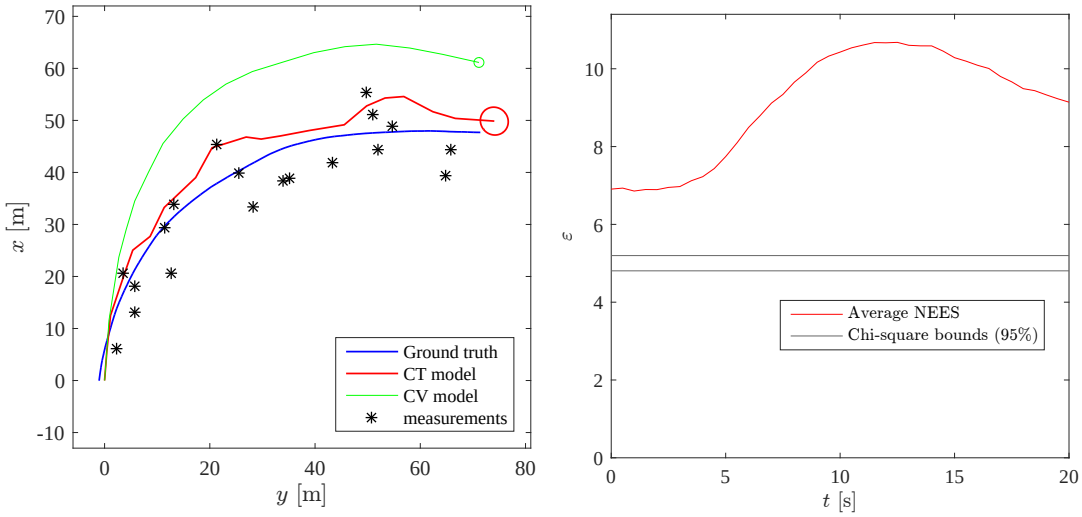


Figure 5.1: Filtering results for a single simulation of the CT scenario and consistency analysis from 1000 Monte-Carlo simulations.

Despite the observed improvements, caution is mandatory. By running 1000 Monte-Carlo

Table 5.2: Parameter values used in the Van der Pol example

Symbol	Description	Value
σ_a	Plant noise standard deviation	0.1
σ_z	Measurement noise standard deviation	0.1
T	Discretization time interval	0.01 s
T	Time interval between measurements	1 s

simulations, we can investigate whether the EKF actually attains ideal consistency properties. The answer, provided by the right-hand-side of Figure 5.1, is ultimately negative. The ANEES of the full 5-dimensional state vector of the CT-EKF is up to twice as large as it should be. While this is not very severe inconsistency, it could make a tracking system more susceptible to track loss. For other choices of the system parameters, the inconsistency problem could be better or worse. ∎

∎ **Example 5.2 — The Van der Pol oscillator.** The archtypical nonlinear system in deterministic system theory is the Van der Pol oscillator. It is similar to a standard linear mass-spring-damper system, that also has a negative damping that comes into play when the state vector approaches its equilibrium point at the origin. A stochastic version of this system, driven by white noise, can be written as $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + \mathbf{G}\mathbf{n}$ where

$$\mathbf{f}(\mathbf{x}) = \begin{bmatrix} x_2 \\ \mu(1-x_1^2)x_2 - x_1 \end{bmatrix}, \ \mathbf{G} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \ \mathbf{n} \sim \mathcal{N}(0, \mathbf{D}\delta(t-\tau)) \tag{5.19}$$

where $\mathbf{D}$ is the (scalar) covariance of the continuous-time white noise process. Let us furthermore assume that only the second state is measured, so the measurement model becomes $\mathbf{z} = \mathbf{H}\mathbf{x} + \mathbf{w}$ where

$$\mathbf{H} = \begin{bmatrix} 0 & 1 \end{bmatrix}, \ \mathbf{w} \sim \mathcal{N}(0, R\delta_{kl}) \tag{5.20}$$

and where R is the (scalar) discrete-time measurement covariance. The first challenge that we encounter in this example is that we are given a *continuous-time* nonlinear plant model, while our formulation of the EKF in Algorithm 2 presumes a *discrete-time* nonlinear plant model. Unfortunately, discretization of arbitrary nonlinear systems do not lead to closed-form solutions. In the EKF we must then devise approximations both for prediction of the state estimate and its covariance. The former is done by any kind of numerical ODE solver, such as Euler's method or your favourite Runge-Kutta method. For the latter, one possibility is to proceed as closely as possible to how we would do this for a linear system, by evaluating the transition matrix for the linearized system and the propagating the covariance using the standard Kalman filter approach for this transition matrix. The system matrix and the plant noise matrix of the linearized model are

$$\mathbf{A} = \begin{bmatrix} 0 & 1 \\ -1-2\mu xy & \mu(1-x^2) \end{bmatrix}, \ \mathbf{G} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \tag{5.21}$$

Over the time interval $T = t_k - t_{k-1}$ we find the corresponding transition matrix $\mathbf{F}$ and the discrete time plant noise covariance $\mathbf{Q}$ according to the standard discretization formulas in (4.59) and (4.61). Then the covariance prediction is given by the standard formula $\mathbf{P}_{k|k-1} = \mathbf{F}\mathbf{P}_{k-1}\mathbf{F}^\mathsf{T} + \mathbf{Q}$. This completes the specification of the EKF for the Van der Pol example. In addition some model parameters and simulation setup parameters must be decided to implement a simulation of this system. These are provided in Table 5.2. ∎

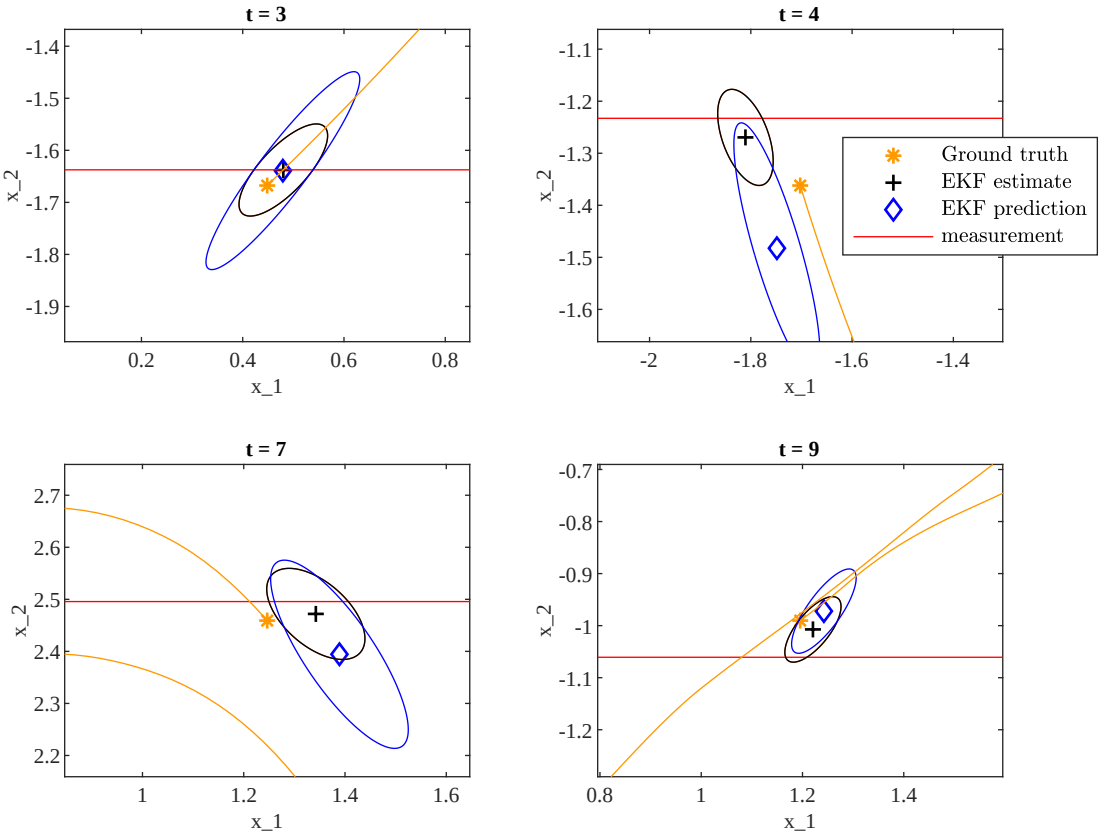


Figure 5.2: Snapshots of EKF outputs for the Van der Pol example.

The output from the EKF in this example is visualized in Figure 5.2 for 4 different time steps. We see that both the predicted and the updated covariance ellipse tend to be aligned with the trajectory of the state vector. This is related to the very strong predictive power inherent in this system: It is well known that the Van der Pol oscillator tends towards a limit cycle. This information is not used in the measurement update, but it does indirectly affect the development of the covariance during the prediction.

5.2 Particle Filters

A key challenge in nonlinear filtering is that, even if all the noise processes are Gaussian, the nonlinear transformations quickly turn the posterior density into something non-Gaussian. Exact solutions are then very unlikely to exist. If the posterior indeed does not resemble a Gaussian at all, then we have little reason to expect that an EKF will work well, and we may desire a more flexible technique that can represent arbitrary posteriors with finite computational resources.

Possible solutions to this problem of non-Gaussian filtering tend to belong to one of three categories. The first category consists of grid-based approaches. For problems in 1 or 2 dimensions it may be feasible to evaluate the posterior density in a brute force manner over a grid with a suitable resolution. For higher dimensions this quickly becomes computationally infeasible. The second category consist of approaches based on random sampling. Instead of placing the samples in a pre-defined grid, we may draw them from a probability distribution which tends to place them where the bulk of the posterior density is likely to be. The third category consists of methods that aim to represent the posterior as a sum of simpler functions, typically Gaussians. In this section

we will explore particle filters, which is the most popular approach from the second category of non-Gaussian filters.

The theory of particle filters is founded upon three main ideas. These can be summarized as *1)* Monte-Carlo integration and importance sampling, *2)* Sampling of trajectories and *3)* Resampling. Our development of the fundamental theory of particle filtering will be organized according to these ideas in the following three subsections, before we present the simplest possible particle filter in Section 5.2.4. More advanced particle filter methods follow in Sections 5.2.5 - 5.2.7.

5.2.1 Idea 1: Monte-Carlo integration and importance sampling

To begin, recall that interesting properties are often extracted from pdf's by means of integration. For example, the expectation of $\mathbf{x}$ under the pdf $\pi(\mathbf{x})$ is given by the integral $\int \mathbf{x}\pi(\mathbf{x})\,\mathrm{d}\mathbf{x}$. For an arbitrary pdf $\pi(\mathbf{x})$ we may easily encounter a situation where this integral has no closed form solution. Does that mean that we have no means of evaluating the expectation of $\mathbf{x}$? Not necessarily. If we are able to produce a large number of independent samples $\mathbf{x}^i$ from $\pi(\mathbf{x})$ then we can simply approximate the integral as the average of these samples.

This can be generalized to a situation where we want to evaluate an integral of the form

$$I = \int \mathbf{f}(\mathbf{x})\,\pi(\mathbf{x})\,\mathrm{d}\mathbf{x}. \tag{5.22}$$

Again, we draw samples $\mathbf{x}^i$ from $\pi(\mathbf{x})$, and approximate the integral as the average of $\mathbf{f}(\mathbf{x}^i)$. In more mathematical terms, we let $\{\mathbf{x}^i\}_{i=1}^N$ be N i.i.d. samples from $\pi(\cdot)$, and we define the corresponding sample mean of $\mathbf{f}(\mathbf{x})$ as

$$I_N = \frac{1}{N}\sum_{i=1}^{N}\mathbf{f}(\mathbf{x}^i). \tag{5.23}$$

Then we can rest assured that

$$I_N \longrightarrow I \quad \text{as} \quad N \longrightarrow \infty. \tag{5.24}$$

This technique for evaluating integrals is known as importance sampling, and $\pi(\mathbf{x})$ is called an importance density.

■ **Example 5.3 — Integral evalutated by importance sampling.** Consider the function displaye in Figure 5.3. This function is given by the expression

$$f(x) = 0.7\mathcal{N}(x;1,0.1) + 0.3\mathcal{N}(x;2.8,0.9). \tag{5.25}$$

Since it is a sum of two pdfs, weighted with numbers that sum to one, it is easy to verify that the integral $\int f(x)\mathrm{d}x$ indeed is one. But let us pretend that we are not in possession of this knowledge. We want to evaluate the integral by means of importance sampling. We consider four possible importance densities:

$$\begin{aligned}\pi_1(x) &= \mathcal{N}(x;1,0.1)\\ \pi_2(x) &= \mathcal{N}(x;2.8,0.9)\\ \pi_3(x) &= \mathcal{N}(x;1.54,0.34)\\ \pi_4(x) &= f(x).\end{aligned}$$

We recognize importance densities 1 and 2 as the Gaussians that make up the summands in (5.25). Neither of these are good representatives for the overall shape of $f(x)$. The third importance density is a Gaussian that is constructed in order to provide a better representation of the shape of $f(x)$ in

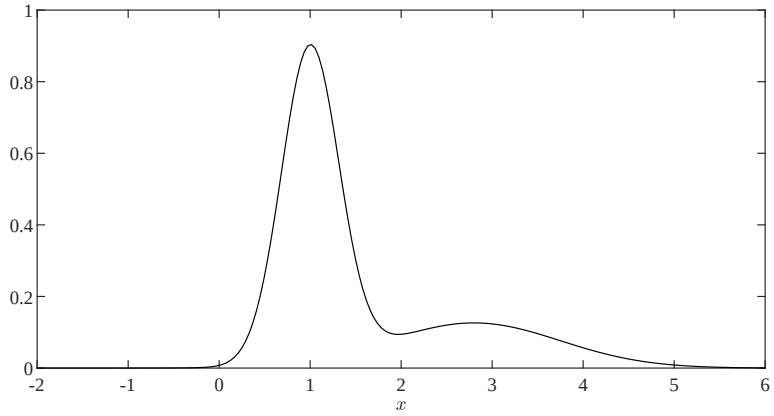


Figure 5.3: Function to be integrated by means of importance sampling.

Table 5.3: Results of importance sampling for the integral of (5.3)

Scheme	$N = 10$	$N = 100$	$N = 10^3$	$N = 10^4$	$N = 10^5$
π_1	0.7256	0.7432	0.7806	0.7710	0.8183
π_2	2.2569	0.9333	0.8900	1.0114	1.0091
π_3	1.0048	1.0164	0.9883	1.0100	0.9995
π_4	1.0000	1.0000	1.0000	1.0000	1.0000

the following sense: It has the same expectation and covariance as $f(x)$ would have if interpreted as a pdf. This is known as moment-matching. Finally, since $f(x)$ actually is a pdf we can also use $f(x)$ itself as an importance density. To use importance sampling, we calculate sample averages of the form

$$\frac{1}{N}\sum_{i=1}^{N}\frac{f(x^i)}{\pi_k(x^i)},\ k \in \{1,2,3,4\}$$

with samples from $\pi_1(\cdot),\ldots,\pi_4(\cdot)$, respectively. Let us then take a look at how the accuracy of the different integration schemes depend on the number N of samples. In Table 5.3 we see some results from single trials of the integration schemes for different values of of N. The first scheme does not perform very well. Even for $N = 10^5$ it is not capable of giving an accurate evaluation. Also notice that it *always* underestimates the integral. The reason for that is that $\pi_1(x)$ is narrower than $f(x)$. Consequently, it is difficult for it to place samples at the plateau between $x = 2$ and $x = 4$, and it fails to take this part of the integrand properly into account. The second scheme uses the widest of the two Gaussians involved in $f(x)$, and consequently it does manage to place samples everywhere where the integrand differs significantly from zero. We see a significant improvement in its performance as N increases. The third scheme appears to have very good performance already for low values of N. Because it is better tailored to the overal structure of $f(x)$ than the second scheme, it manages to place its samples more strategically. Finally, the fourth scheme is able to find the exact value of the integral no matter what N is, because once its proposal density is factored out of the integrand, all we are left with is the constant one.

There are two take-home lessons from this example. First, it is clearly desirable with a proposal density that is a good approximation of the integrand. Second, it is important that the proposal density places samples wherever the integrand is has significant mass, and therefore it is better with a proposal density that is a little bit too wide than a proposal density that is a little bit too narrow.

Keep in mind that these conclusions were made only based on a single Monte-Carlo simulation. To state anything with certainty a larger number (typically hundreds or thousands) of Monte-Carlo simulations may be needed. We save that for the next example. ∎

∎ **Example 5.4 — 2-dimensional integral evalutated by importance sampling.** In this example we want to evaluate the integral $\int_D \mathcal{N}(\mathbf{x}; \mathbf{a}, \mathbf{A})\mathrm{d}\mathbf{x}$ the region of integration is $D = (-\infty, \infty) \times [0, 1]$, and where

$$\mathbf{a} = \begin{bmatrix} 0 \\ -1 \end{bmatrix}, \ \mathbf{A} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}.$$

This scenario is illustrated in Figure 5.4. It is motivated by the problem of calculating collision probabilities in a collision avoidance system [83]. Let us consider two possible importance densities:

$$\pi_1(\mathbf{x}) = \mathcal{N}(\mathbf{x}; \mathbf{a}, \mathbf{A})$$
$$\pi_2(\mathbf{x}) = \mathcal{N}(x_1; 0, 1)\mathrm{Uniform}(x_2; [0, 1])$$

The accuracy of the two proposal densities is evaluated for different sample numbers over 1000

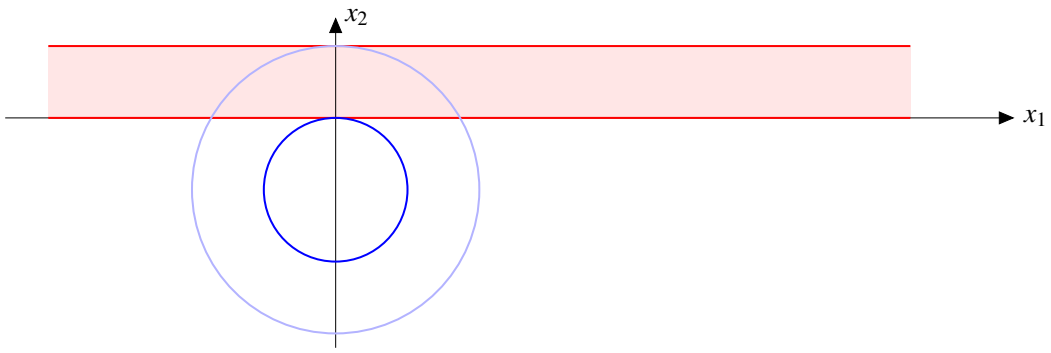


Figure 5.4: Illustration of the problem to be solved in Example 5.4. The integration domain is the red strip, while the integrand is the Gaussian whose first and second covariance ellipses are shown in blue.

independent Monte-Carlo simulations. In Figure 5.5 we see the relative RMSE of the two integration schemes, that is, RMSE divided by the exact value of the integral, which is 0.1359. Overall, the integration scheme using $\pi_1(\mathbf{x})$ has 10 times larger errors than the integration scheme using $\pi_2(\mathbf{x})$. To reach a relative accuracy of about 0.01 we need about 1000 samples with $\pi_2(\mathbf{x})$, while we need close to 10^5 samples with $\pi_1(\mathbf{x})$. The reason is that $\pi_1(\mathbf{x})$ is much wider than it should be in the x_2-direction in order to place its samples within the red strip. ∎

So far we have explored importance sampling for integrals where we have a closed-form integrand. The situation that we soon shall encounter in particle filtering is slightly more complex. In a particle filter we are again dealing with an integral of the form $I = \int \mathbf{f}(\mathbf{x})\, \pi(\mathbf{x})\, \mathrm{d}\mathbf{x}$, but in this case we are only able to express $\pi(\mathbf{x})$ indirectly. More to the point, we will not be able to sample from $\pi(\mathbf{x})$ itself. The solution is to introduce yet another proposal density, called $q(\mathbf{x})$, so that the integral also can be written as

$$I = \int \mathbf{f}(\mathbf{x}) \frac{\pi(\mathbf{x})}{q(\mathbf{x})} q(\mathbf{x})\, \mathrm{d}\mathbf{x}. \tag{5.26}$$

We introduce the so-called importance weights

$$\tilde{w}(\mathbf{x}^i) = \frac{\pi(\mathbf{x}^i)}{q(\mathbf{x}^i)} \tag{5.27}$$

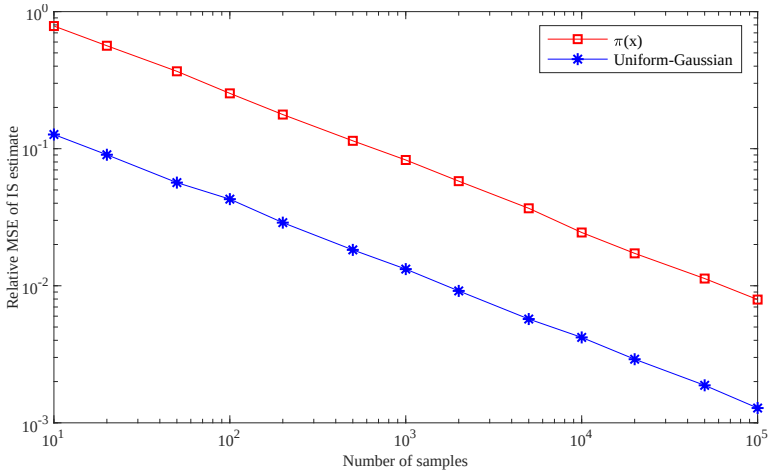


Figure 5.5: Accuracy of Monte-Carlo integration in Example 5.4.

and approximate the integral by the sample mean

$$I_N = \frac{1}{N}\sum_{i=1}^{N} \mathbf{f}(\mathbf{x}^i)\tilde{w}(\mathbf{x}^i). \tag{5.28}$$

Thanks to (5.26) we can again rest assured that $I_N \longrightarrow I$ as $N \longrightarrow \infty$. In particle filtering applications, the pdf $\pi(\mathbf{x})$ will typically only be known up to proportionality. We can then use the normalized importance weights and still get the correct result (see Exercise 5.2):

$$w(\mathbf{x}^i) = \frac{\tilde{w}(\mathbf{x}^i)}{\sum_{j=1}^{N} \tilde{w}(\mathbf{x}^j)}. \tag{5.29}$$

We saw in the examples that the choice of the importance density is critical to the efficiency of Monte-Carlo integration. It is desirable to choose a $q(\mathbf{x})$ that is a good approximation of $\pi(\mathbf{x})$. In general, our choice of $q(\mathbf{x})$ will typically be either narrower or wider than $\pi(\mathbf{x})$. The former is generally worse than the latter. A central concept here is *support*: The support of a function $f(\mathbf{x})$ is the set of all $\mathbf{x}$ such that $f(\mathbf{x}) > 0$. For importance sampling to work at all, the support of $q(\mathbf{x})$ must include the entire support of $\pi(\mathbf{x})$. This is equivalent to demanding that $\pi(\mathbf{x})/q(\mathbf{x})$ is upper bounded. Even if this is satisfied, importance sampling may still be horribly inefficient if there is a significant difference in where $\pi(\mathbf{x})$ has the bulk of its probability mass and where $q(\mathbf{x})$ has its probability mass.

5.2.2 Idea 2: Sampling of trajectories

To use importance sampling in recursive Bayesian estimation, we need mechanisms to represent the posterior $p(\mathbf{x}_k \,|\, \mathbf{z}_{1:k})$ in terms of samples from an importance density $q(\mathbf{x}_k \,|\, \mathbf{z}_{1:k})$. The challenge here is that we do not have any closed form expression for $p(\mathbf{x}_k \,|\, \mathbf{z}_{1:k})$ available to insert in the weight formula (5.27). We must therefore come up with a sampling scheme that allows us to only use known densities, i.e. $p(\mathbf{z}_k \,|\, \mathbf{x}_k)$ and $p(\mathbf{x}_k \,|\, \mathbf{x}_{k-1})$, in the weight evaluation.

The solution that a conventional particle filter uses to overcome this challenge is to sample not only the current state $\mathbf{x}_k$, but actually the entire trajectory $\mathbf{x}_{1:k}$. This works as follows. First N samples of the initial states $\mathbf{x}_1$ are drawn, and their weights are updated according to how well they explain the first measurement $\mathbf{z}_1$. Then one sample for each of these are drawn of the next state vector $\mathbf{x}_2$ and weights are calculated. As the particle filter iterates through the time steps, each sample thus becomes a possible trajectory of the system.

Delving into the mathematics, we notice that the full posterior of the trajectory $\mathbf{x}_{1:k}$ is given by

$$p(\mathbf{x}_{1:k}|\mathbf{z}_{1:k}) \propto p(\mathbf{z}_k|\mathbf{x}_k)\, p(\mathbf{x}_k|\mathbf{x}_{k-1})\, p(\mathbf{x}_{1:k-1}|\mathbf{z}_{1:k-1}). \tag{5.30}$$

We assume that the old posterior $p(\mathbf{x}_{1:k-1}\,|\,\mathbf{z}_{1:k-1})$ is given by a set of particles $\mathbf{x}^i_{1:k-1}$, which are drawn according to a proposal density $q(\mathbf{x}_{1:k-1}\,|\,\mathbf{z}_{1:k-1})$ and with weights

$$w^i_{k-1} \propto \frac{p(\mathbf{x}^i_{1:k-1}|\mathbf{z}_{1:k-1})}{q(\mathbf{x}^i_{1:k-1}|\mathbf{z}_{1:k-1})}. \tag{5.31}$$

The key step in the derivation is then to assume that the new proposal density $q(\mathbf{x}_{1:k}\,|\,\mathbf{z}_{1:k})$ is chosen in such a way that it factorizes as follows:

$$q(\mathbf{x}_{1:k}|\mathbf{z}_{1:k}) = q(\mathbf{x}_k|\mathbf{x}_{1:k-1},\mathbf{z}_{1:k-1})q(\mathbf{x}_{1:k-1}|\mathbf{z}_{1:k-1}) \tag{5.32}$$

If we combine the last three equations we arrive at the following expression for the current importance weights

$$\begin{aligned} w^i_k &\propto \frac{p(\mathbf{x}^i_{1:k}|\mathbf{z}_{1:k})}{q(\mathbf{x}^i_{1:k}|\mathbf{z}_{1:k})} \\ &\propto \frac{p(\mathbf{z}_k\,|\,\mathbf{x}^i_k)p(\mathbf{x}^i_k\,|\,\mathbf{x}^i_{k-1})p(\mathbf{x}^i_{1:k-1}\,|\,\mathbf{z}_{1:k-1})}{q(\mathbf{x}^i_k\,|\,\mathbf{x}_{1:k-1},\mathbf{z}_{1:k-1})q(\mathbf{x}^i_{1:k-1}|\mathbf{z}_{1:k-1})} \\ &\propto \frac{p(\mathbf{z}_k\,|\,\mathbf{x}^i_k)p(\mathbf{x}^i_k\,|\,\mathbf{x}^i_{k-1})}{q(\mathbf{x}^i_k\,|\,\mathbf{x}_{1:k-1},\mathbf{z}_{1:k-1})} w^i_{k-1} \end{aligned} \tag{5.33}$$

The main take-home message from (5.33) is that we have a *recursive* formula for the weight updates. Without that, we would have needed to evaluate the trajectory pdf's $p(\mathbf{x}_{1:k}|\mathbf{z}_{1:k})$ and $q(\mathbf{x}_{1:k}|\mathbf{z}_{1:k})$ every time a new measurement is being processed.

In the standard Bayesian filtering problem we rely on the Markov assumption $p(\mathbf{x}_k\,|\,\mathbf{x}_{1:k-1}) = p(\mathbf{x}_k\,|\,\mathbf{x}_{k-1})$. That is, we assume that earlier states $\mathbf{x}_1,\ldots,\mathbf{x}_{k-2}$ carry no information about $\mathbf{x}_k$ that is not carried by $\mathbf{x}_{k-1}$. Under this assumption there should not be much to gain from invoking earlier states or measurements in the proposal density, and it makes sense to restrict it to the form

$$q(\mathbf{x}_k|\mathbf{x}_{1:k-1},\mathbf{z}_{1:k}) = q(\mathbf{x}_k|\mathbf{x}_{k-1},\mathbf{z}_k). \tag{5.34}$$

With this proposal density, the weight update becomes

$$w^i_k \propto w^i_{k-1} \frac{p(\mathbf{z}_k|\mathbf{x}^i_k)\, p(\mathbf{x}^i_k|\mathbf{x}^i_{k-1})}{q(\mathbf{x}^i_k|\mathbf{x}^i_{k-1},\mathbf{z}_k)} \tag{5.35}$$

While further specification of the proposal density differs between the many variations of the particle filter, all particle filters known to the author is based on the general framework specified by (5.34) and (5.35).

While (5.34) and (5.35) really describe a methodology for importance sampling of trajectories, we are in filtering (as opposed to, e.g., smoothing) only interested in the marginal posterior distribution at the current time step. It is given by

$$p(\mathbf{x}_k\,|\,\mathbf{z}_{1:k}) = \int p(\mathbf{x}_{1:k}\,|\,\mathbf{z}_{1:k})\,\mathrm{d}\mathbf{x}_{1:k-1}. \tag{5.36}$$

If $p(\mathbf{x}_{1:k}\,|\,\mathbf{z}_{1:k})$ is represented by a collection of samples, then the marginalization boils down to simply forgetting the first $k-1$ state vectors in every sample. Since this is what we do in practice, it may on the surface seem like we only sample the most recent state vector in a particle filter. But the derivations of this section shows that this is an illusion.

5.2.3 Idea 3: Resampling

Since a particle filter really samples trajectories and not just state vectors, the dimension of its samples increases during every iteration. This is problematic, because as the sample space becomes larger and larger, it becomes more and more difficult for the proposal density to hit the right regions. A symptom of this problem is that the variance of the weights will increase until one particle has all the probability mass. A proof that this indeed will happen can be found in [52]. An intuitive understanding can be gained from recalling that the logarithm of any Gaussian is a quadratic form. Assume for this little thought exercise that the pdf that we want to sample from is approximately Gaussian. If the dimension of the state space increases, and it becomes more difficult to place samples near the peak of the Gaussian, then the samples will tend to end up further away where the quadratic form is steeper. It is illustrated in Figure 5.6 how the rightmost couple of samples x_3 and x_4 will get very different pdf-values, while the difference between the pdf-values for the samples near the peak (x_1 and x_2) is negligible.

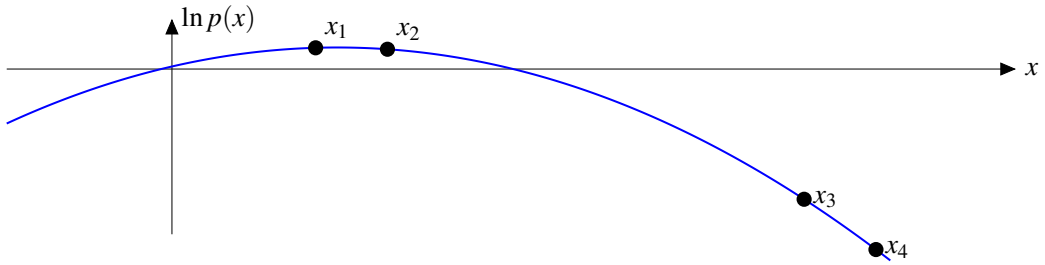


Figure 5.6: Illustration of how the ratio of pdf-values from a Gaussian increases for samples drawn further away from the expectation. A particle filter attempting to estimate this posterior will get more and more imbalanced weights the further away the particle cloud is from the maximum.

The problem that only one or a few of the particles take all the probability mass is known as *degeneracy*. It can be monitored by a quantity known as the effective sample size:

$$\hat{N}_{\text{eff}} = \frac{1}{\sum_{i=1}^{N} (w_k^i)^2}. \tag{5.37}$$

We are generally happy if $\hat{N}_{\text{eff}} \approx N$. On the other hand, if $\hat{N}_{\text{eff}} \to 1$ then we have degeneracy. It is therefore advisable to do something if $\hat{N}_{\text{eff}}$ goes below some pre-determined threshold.

The solution to this problem is to *resample* the particles. We generate a new particle cloud, where each new particle is drawn from the collection of old particles, with probabilities equal to the old particle weights. This way, we get several copies of the particles with high weights, while particles with lower weights get fewer or no copies. All the new particles get uniform weights, and the degeneracy problem is thus solved.

The most straightforward approach to resampling is to use *cdf inversion* as described in Section 2.2.4. How this works for a particle filter is illustrated in Figure 5.7. First, we construct the cdf of the old particle cloud. This will be a staircase function with jumps for every integer j between 1 and N. For each new particle (indexed by i) we draw a random number u^i uniformly distributed between 0 and 1. Then we determine which step covers u^i. The corresponding particle then becomes the new particle number i. After repeating this procedure N times, a new particle cloud has been generated.

A naive implementation of resampling by cdf inversion would work in $O(N^2)$ time. For each new particle, we find the corresponding old particle. It is possible to reduce the complexity to $O(N \log N)$ by sorting the random numbers u^i first. Further reduction to $O(N)$ complexity is possible by utilizing algorithms based on order-statistics.

A more convenient approach to resampling is *systematic resampling*. In this approach, only a single random number is drawn, and then new particle copies are generated by moving along

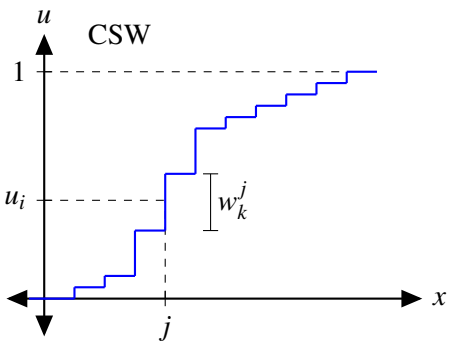


Figure 5.7: Resampling by CDF inversion.

the cumulative sum of weights, without any sorting needed. A pseudo-code description of the algorithm is given in Algorithm 3. The scrambling in Line 13 of the pseudocode is not strictly needed, but recommended to ensure that everything remains as random as possible.

Algorithm 3 Systematic resampling

```
1:  procedure ROULETTESYSTEMATIC(weights, N)
2:      cumweights ← cumsum(weights)
3:      indicesout ← zeros(N, 1)
4:      noise ← rand(1, 1)/N
5:      i ← 1
6:      for j ← 1 : N do
7:          u_j ← (j − 1)/N
8:          while u_j > cumweights[i] do
9:              i ← i + 1
10:         end while
11:         indicesout[j] ← i
12:     end for
13:     indicesout ← indicesout(randperm(size(indicesout, 1)))
14:     return indicesOut
15: end procedure
```

5.2.4 The SIR filter and practical implementation

We now have all the building blocks in place to design our first particle filter. The main question when designing a particle filter is always: What should our proposal density $q(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathbf{z}_k)$ be? The simplest answer to that question is that it can be given by the process model:

$$q(\mathbf{x}_k^i|\mathbf{x}_{k-1}^i, \mathbf{z}_k) = p(\mathbf{x}_k^i \,|\, \mathbf{x}_{k-1}^i). \tag{5.38}$$

For this proposal density, the weight update formula becomes

$$w_k^i =\propto w_{k-1}^i \frac{p(\mathbf{z}_k|\mathbf{x}_k^i)\, p(\mathbf{x}_k^i|\mathbf{x}_{k-1}^i)}{p(\mathbf{x}_k \,|\, \mathbf{x}_{k-1})} = w_{k-1}^i p(\mathbf{z}_k|\mathbf{x}_k^i) \tag{5.39}$$

The particle filter that results is known as the Sequential Importance-sampling-Resampling filter, abbreviated as the SIR filter. Before we delve into our first example of a particle filter, an important remark is in order regarding how the weight update (5.39) is implemented in practice. The likelihood

$p(\mathbf{z}_k|\mathbf{x}_k^i)$ can vary tremendously in value, and is very likely to encounter situations where all the likelihood values become indistinguishable from zero due to the limited numerical precision of any computer. The solution to this problem is to work with the logarithms of $p(\mathbf{z}_k|\mathbf{x}_k^i)$ and w_k^i. The product in (5.39) then becomes a sum. This may, however, not be sufficient. To carry out the resampling step, we have to calculate the normalized weights, and before we can do this we have to exponentiate the logarithmic weights, which again carries a danger of numerical underflow. To ensure that at least the largest weights give meaningful numbers after this exponentiation, we can rescale the logarithmic weights before the exponentiation by adding a constant to all the logarithmic weights so that the largest logarithmic weight becomes zero. The overall workflow of the SIR filter is summarized by the block diagram in Figure 5.8.

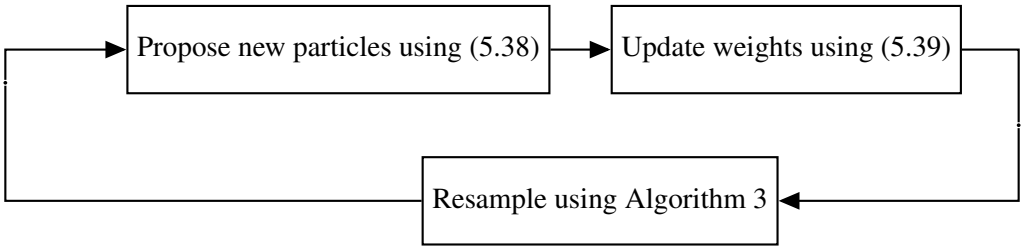


Figure 5.8: Workflow of the SIR filter. Each estimation cycle consists of three steps.

■ **Example 5.5 — SIR filter for the CV model.** Before we consider any nonlinear applications, it is a good idea to study how well a particle filter is able to deal with a straightforward linear scenario. Therefore we implement the SIR filter for the standard CV model. We use the same values for σ_z and T as were used for the CV model in Table 5.1. We investigate the two values for the process noise strength σ_a that were considered in Example 4.11, that is $\sigma_a = 0.5$ and $\sigma_a = 0.05$. We generate the simulated trajectories from initial state and covariance given by

$$\mathbf{x}_0 = \begin{bmatrix} 0 \\ 0 \\ 5 \\ 0 \end{bmatrix}, \quad \mathbf{P}_0 = \begin{bmatrix} 25 & 0 & 0 & 0 \\ 0 & 25 & 0 & 0 \\ 0 & 0 & 25 & 0 \\ 0 & 0 & 0 & 25 \end{bmatrix}.$$

The simulation models and the filter models are identical. The only remaining tuning constant is the number of particles, so let us study how this choice affects the performance of the SIR filter, in addition to the process noise covariance.

In Figure 5.9 we compare the performance of the SIR filter and the Kalman filter in the high noise scenario $\sigma_a = 0.5$ when 10, 100, 1000, 10000 and 100000 particles are used. We see that all the particle filters with more than 100 particles have performance on par with the Kalman filter. However, when only 10 particles are used, the performance is not acceptable.

One would expect that performance should increase when the process noise is lowered to 0.05. However, Figure 5.10 reveals a very different picture. In this case, we need more than 10000 particles to get performance comparable to the Kalman filter. With 1000 particles the RMSE is several times larger for the particle filter than for the Kalman filter. Further reductions in the value of σ_a only makes the performance even worse.

If we dive deeper into these bewildering results, it is revealed that the SIR filter in the low-noise scenario often diverges because none of its particles are good enough. Once divergence happens, it has no mechanism to restore its performance. In Figure 5.11 we see a case study of how this happens for a Monte-Carlo simulation with 1000 particles. After 10 seconds (20 time steps) the particle cloud has collapsed into a collection of clusters, of which none really manages to capture the true posterior given by the Kalman filter. After 20 seconds only one of these clusters remain, and it drifts slowly and indefinitely away from the true state for future time steps. ■

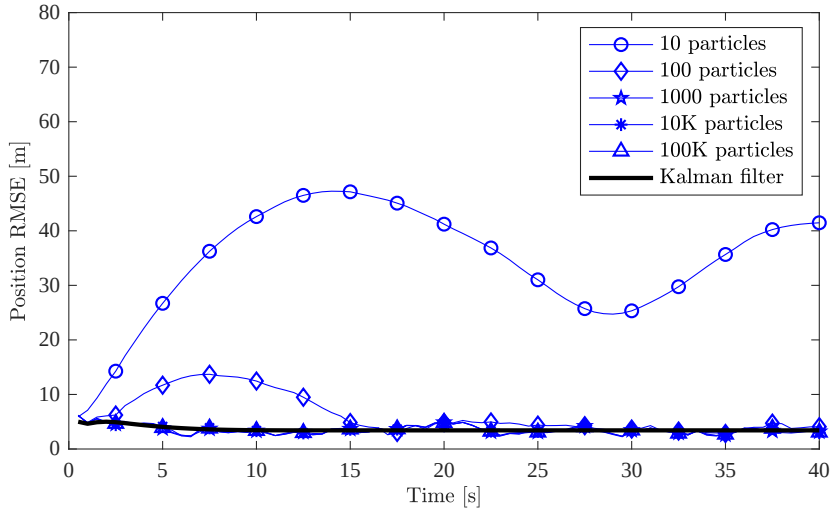


Figure 5.9: Comparison of SIR filter and Kalman filter for the CV model when $\sigma_a = 0.5$

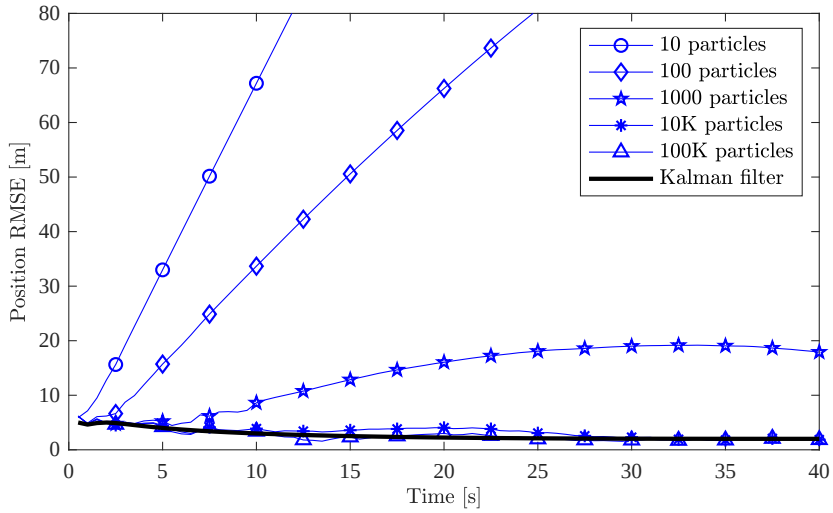


Figure 5.10: Comparison of SIR filter and Kalman filter for the CV model when $\sigma_a = 0.05$

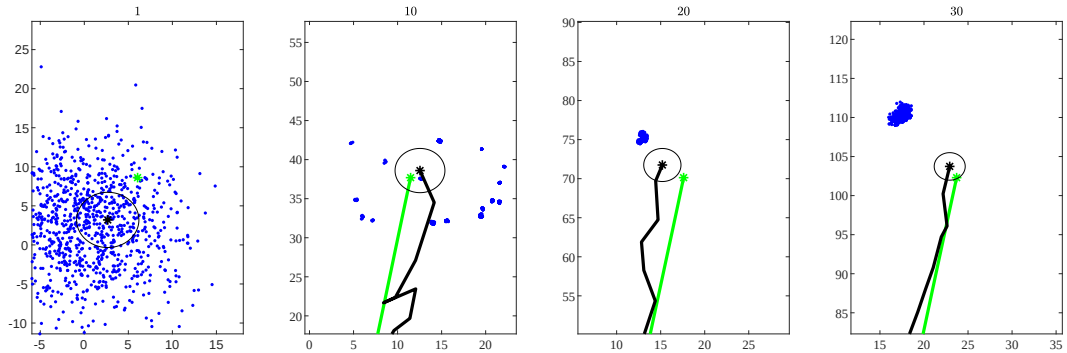


Figure 5.11: Development of particle depletion when $\sigma_a = 0.05$.

The effect seen in Figure 5.11 is known as *particle depletion*. It is clearly a reason to be cautious with particle filters. We can always circumvent the problem by increasing the process noise in the filter model. But this is cheating. If we are not able to present the particle filter with an honest model of the scenario, but must tell it that the scenario is noisier than it actually is, then any uncertainty statistics from the particle filter will be fundamentally unreliable.

The main take-home message from the example is that it is not trivial to design a particle filter that works well. In the remainder of this section on particle filters we will cover three standard approaches to improve particle filter performance: Data-dependent proposal densities, regularization and Rao-Blackwellization. These can very well be combined in the same filter. Among these techniques, only regularization does actually solve the collapse problem seen in Figure 5.11. This is kind of disappointing, as regularization is the least elegant and most heuristic of the three techniques.

5.2.5 Designing a good proposal density

In our derivation of the mathematics of the particle filter in Section 5.2.2 we arrived at a proposal density of the general form $q(\mathbf{x}_k|\mathbf{x}_{1:k-1},\mathbf{z}_{1:k}) = q(\mathbf{x}_k|\mathbf{x}_{k-1},\mathbf{z}_k)$. That is, given a previous particle $\mathbf{x}^i_{1:k-1}$, we draw its child particle $\mathbf{x}^i_{1:k}$ from a pdf that is given by $\mathbf{x}^i_{k-1}$ and $\mathbf{z}_k$. The SIR filter conforms to this scheme, but only $\mathbf{x}^i_{k-1}$ was used in its proposal density, while $\mathbf{z}_k$ was ignored. This is clearly suboptimal.

The *optimal proposal density*, given that $\mathbf{x}^i_{k-1}$ is the previous state, is given by

$$q(\mathbf{x}_k\,|\,\mathbf{x}^i_{k-1},\mathbf{z}_k)_{\text{opt}} = p(\mathbf{x}_k\,|\,\mathbf{x}^i_{k-1},\mathbf{z}_k) \propto p(\mathbf{z}_k\,|\,\mathbf{x}^i_{k-1},\mathbf{x}_k)\,p(\mathbf{x}_k\,|\,\mathbf{x}^i_{k-1}). \tag{5.40}$$

In other words, it is the posterior density of $\mathbf{x}_k$ given that $\mathbf{x}_{k-1}$ had the value $\mathbf{x}^i_{k-1}$. The weight update in (5.35) now becomes

$$w^i_k \propto w^i_{k-1} p(\mathbf{z}_k\,|\,\mathbf{x}^i_{k-1}) = w^i_{k-1} \int p(\mathbf{z}_k\,|\,\mathbf{x}_k) p(\mathbf{x}_k\,|\,\mathbf{x}^i_{k-1})\,\mathrm{d}\mathbf{x}_k. \tag{5.41}$$

With this scheme we are likely to encounter two difficulties. First, the integral in (5.41) will probably not have a closed-form solution if either $p(\mathbf{z}_k\,|\,\mathbf{x}_k)$ or $p(\mathbf{x}_k\,|\,\mathbf{x}^i_{k-1})$ is non-Gaussian or nonlinear[2], which presumably is the case. Second, it may be difficult to sample from $p(\mathbf{x}_k\,|\,\mathbf{x}^i_{k-1},\mathbf{z}_k)$.

Nevertheless, special circumstances exist where the optimal proposal scheme can be used. More precisely, it can be used if the measurement model is linear and both plant noise and measurement noise are Gaussian. In other words, a prerequisite for using the optimal proposal density without approximations is that nonlinearities only occur in the process model. Such a model is of the form

$$\begin{aligned} \mathbf{x}_k &= \boldsymbol{f}(\mathbf{x}_{k-1}) + \mathbf{v}_{k-1} \\ \mathbf{z}_k &= \mathbf{H}\mathbf{x}_k + \mathbf{w}_k \end{aligned} \tag{5.42}$$

where $\mathbf{v}_{k-1}$ and $\mathbf{w}_k$ are mutually independent zero-mean white Gaussian sequences with covariances $\mathbf{Q}$ and $\mathbf{R}$. For particle number i, let us define

$$\begin{aligned} \mathbf{a}^i_k &= \boldsymbol{f}(\mathbf{x}^i_{k-1}) + \boldsymbol{\Sigma}\mathbf{H}^\mathsf{T}\mathbf{R}^{-1}(\mathbf{z}_k - \mathbf{b}^i_k) \\ \boldsymbol{\Sigma} &= \mathbf{Q} - \mathbf{Q}\mathbf{H}^\mathsf{T}\mathbf{S}^{-1}\mathbf{H}\mathbf{Q} \\ \mathbf{S} &= \mathbf{H}\mathbf{Q}\mathbf{H}^\mathsf{T} + \mathbf{R} \\ \mathbf{b}^i_k &= \mathbf{H}\boldsymbol{f}(\mathbf{x}^i_{k-1}). \end{aligned} \tag{5.43}$$

[2] By stating that $p(\mathbf{z}_k\,|\,\mathbf{x}_k)$ is nonlinear it is mean that the relationship between $\mathbf{z}_k$ and $\mathbf{x}_k$ is nonlinear.

The optimal proposal density is then given by

$$q(\mathbf{x}_k \,|\, \mathbf{x}_{k-1}^i, \mathbf{z}_k)_{\text{opt}} = \mathcal{N}(\mathbf{x}_k \,;\, \mathbf{a}_k^i, \boldsymbol{\Sigma}) \tag{5.44}$$

while the corresponding weight update is given by $w_k^i \propto w_{k-1}^i p(\mathbf{z}_k \,|\, \mathbf{x}_k^i)$ where

$$p(\mathbf{z}_k \,|\, \mathbf{x}_{k-1}) = \mathcal{N}(\mathbf{z}_k \,;\, \mathbf{b}_k^i, \mathbf{S}). \tag{5.45}$$

The proof of (5.44) and (5.45) is left as an exercise to the reader.

In the more general case with nonlinear measurements or non-Gaussian noise we can still attempt to *approximate* the optimal proposal density using closed-form techniques, and this is in general a recommended design strategy. If the nonlinear mappings $\boldsymbol{f}(\mathbf{x}_{k-1})$ and $\mathbf{h}(\mathbf{x}_k)$ of the general model are sufficiently smooth, the most obvious strategy is to use an EKF to obtain this approximation. This is known as a local linearization particle filter. In this approach, the proposal density is of the form

$$q(\mathbf{x}_k^i \,|\, \mathbf{x}_{k-1}^i, \mathbf{z}_k) = \mathcal{N}(\mathbf{x}_k^i \,;\, \hat{\mathbf{x}}_k^i, \hat{\mathbf{P}}_k^i) \tag{5.46}$$

where the vector $\hat{\mathbf{x}}_k^i$ and the matrix $\hat{\mathbf{P}}_k^i$ are the output of an EKF for particle i. More precisely, this EKF takes the previous estimate-covariance pair and uses it to make a prediction at the current time step, after which the measurement $\mathbf{z}_k$ is used to obtain the current pair through a standard EKF update:

$$(\mathbf{x}_{k-1}^i, \hat{\mathbf{P}}_{k-1}^i) \longrightarrow (\bar{\mathbf{x}}_k^i, \bar{\mathbf{P}}_k^i) \longrightarrow (\hat{\mathbf{x}}_k^i, \hat{\mathbf{P}}_k^i).$$

With this scheme, the particle weights are updated according to

$$w_k^i \propto w_{k-1}^i \frac{p(\mathbf{z}_k \,|\, \mathbf{x}_k^i) p(\mathbf{x}_k^i \,|\, \mathbf{x}_{k-1}^i)}{\mathcal{N}(\mathbf{x}_k^i \,;\, \hat{\mathbf{x}}_k^i, \hat{\mathbf{P}}_k^i)}. \tag{5.47}$$

5.2.6 Regularized particle filter

Unfortunately, using an optimal or near-optimal proposal density is often not enough to ensure that a particle filter behaves well, even in low-dimensional spaces. This has to do with the particle depletion phenomenon that we encountered in Example 5.5. The culprit behind this is the resampling step: Resampling inevitably has a cluster-generating effect on the particle cloud. Each particle that survives resampling becomes a collection of new particles. Each particle in one such collection will subsequently evolve independently according to the proposal density. If the proposal density is narrow, these particles may take a long time to spread out and fill the gaps between the collections generated by the resampling, and a cluster pattern can be observed.

In contrast to the fragmented particle cloud seen in Figure 5.11, we know that the true posterior in Example 5.5 is a Gaussian. Based on this, it seems that one possible strategy to mitigate the particle depletion problem could be to break up the clusters by some kind of jittering. If we calculate the sample covariance of the particle cloud we get a measure of how much the particles should be jittered. The regularized particle filter uses this information to supplement the conventional resampling step with an additional jittering step.

A key concept in the jittering step is that of a *kernel density*. The kernel density is used to draw a new value of $\mathbf{x}_k^i$ for each resampled particle. The new value should not be too far from the old one, so that the overall particle cloud retains its shape. But it should still be sufficiently far away that the gaps between the depletion fragments are filled. This trade-off is solved by choosing an optimal bandwidth for the kernel. Different kernel shapes are possible. A so-called Epanechnikov

kernel, which basically is a pdf shaped as a concave parabola, is considered the optimal shape [73]. However, a Gaussian kernel is generally easier to implement, and can be written as

$$k(\mathbf{x}_k \,|\, \mathbf{x}_k^i) = \mathcal{N}(\mathbf{x}_k \,;\, \mathbf{x}_k^i, h^2\hat{\mathbf{P}}) \tag{5.48}$$

where $\hat{\mathbf{P}}$ is the sample covariance of the particle cloud and h is the bandwidth. The sample covariance is given by

$$\hat{\mathbf{x}}_k = \sum_{i=1}^{N} w_k^i \mathbf{x}_k^i \tag{5.49}$$

$$\hat{\mathbf{P}} = \sum_{i=1}^{N} w_k^i (\mathbf{x}_k^i - \hat{\mathbf{x}}_k)(\mathbf{x}_k^i - \hat{\mathbf{x}}_k)^{\mathsf{T}}. \tag{5.50}$$

The optimal bandwidth for the Gaussian kernel is given by

$$h = A \cdot N^{\frac{1}{n+4}} \quad \text{with} \quad A \left(\frac{4}{n+2} \right)^{\frac{1}{n+4}} \tag{5.51}$$

where N is the number of particles and n is the dimension of the state space.

The reason why the regularized particle filter is described as inelegant above is that this step makes it impossible to guarantee that the particles actually do represent the true posterior. For a standard particle filter, one can rest assured that if one just have enough particles, then these particles will be distributed exactly according to the posterior (of course, "enough" may in reality mean billions upon billions). For a regularized particle filter no such convergence result can be established.

5.2.7 Rao-Blackwellization

To make particle filters work with sufficient efficiency to be of practical interest, it is important to utilize whatever knowledge we have about the *structure* of the problem. One such structure of importance is the placement of nonlinearities in the model. A nonlinear system will typically have some state variables that go through nonlinearities, and some state variables that only occur linearly in the process and measurement models. As an example, let us recall the CT model of Example 5.1, and more precisely its discrete-time process model (5.13). In this model, the current state is given by a matrix times the old state plus some process noise. This does indeed sound like a rather linear model. The only reason why it is nonlinear is that the turn-rate appears in that matrix, subject to different trigonometric, and thus nonlinear, functions. In other words, the turn-rate is the only one of the 5 state variables that actually is exposed to nonlinearities. The question therefore arises: Can we use a particle filter to estimate the nonlinear turn-rate state, and leave the remaining states to a Kalman filter? This is known as Rao-Blackwellization.

Since our aim is to understand how a Rao-Blackwellized particle filter (RBPF) works, rather than having an RBPF available for all eventualities, we shall develop the RBPF for a somewhat limited model, that will be sufficient for our purposes. Our model is

$$\mathbf{x}_k^n = \boldsymbol{f}_k(\mathbf{x}_{k-1}^n) + \mathbf{v}_k^n \tag{5.52}$$

$$\mathbf{x}_k^l = \mathbf{A}_k(\mathbf{x}_{k-1}^n)\mathbf{x}_{k-1}^l + \mathbf{v}_k^l \tag{5.53}$$

$$\mathbf{z}_k = \mathbf{h}(\mathbf{x}_k^n) + \mathbf{C}_k(\mathbf{x}_k^n)\mathbf{x}_k^l + \mathbf{w}_k \tag{5.54}$$

where the noises are zero-mean Gaussian with covariances $\mathbf{Q}^n$, $\mathbf{Q}^l$ and $\mathbf{R}$, respectively, and the usual independence assumptions apply. For this model to be feasible it must be possible to partition the state vector $\mathbf{x}_k$ into a nonlinear part $\mathbf{x}_k^n$ and a linear part $\mathbf{x}_k^l$. We see that $\mathbf{x}_k^l$ only occurs in linear

relationships both in the process model and in the measurement model. to separate the linear part from the nonlinear part, we make use of the following factorization of the posterior:

$$p(\mathbf{x}_k^l, \mathbf{x}_{1:k}^n \,|\, \mathbf{z}_{1:k}) = p(\mathbf{x}_k^l \,|\, \mathbf{x}_{1:k}^n, \mathbf{z}_{1:k}) p(\mathbf{x}_{1:k}^n \,|\, \mathbf{z}_{1:k}). \tag{5.55}$$

It is obvious that we can make such a factorization; this result hinges on nothing more than the definition of conditional probability. We have chosen to condition on the trajectory $\mathbf{x}_{1:k}^n$, and not merely on the latest state vector $\mathbf{x}_k^n$, for reasons that will become evident in connection with the following theorem. Let us furthermore assume that the linear part of (5.55) at the previous time step is given by

$$p(\mathbf{x}_{k-1}^l \,|\, \mathbf{x}_{1:k-1}^n, \mathbf{z}_{1:k-1}) = \mathcal{N}(\mathbf{x}_{k-1}^l \,;\, \hat{\mathbf{x}}_{k-1}^l(\mathbf{x}_{1:k-1}^n), \mathbf{P}_{k-1}^l(\mathbf{x}_{1:k-1}^n)). \tag{5.56}$$

In the sequel, we shall suppress the dependencies on $\mathbf{x}_k^n$ and $\mathbf{x}_{1:k-1}^n$, so that we simply can write $\boldsymbol{f}_k^n$, $\mathbf{h}_k^n$, $\mathbf{A}_k$, $\mathbf{C}_k$, $\hat{\mathbf{x}}_{k-1}^l$ and $\mathbf{P}_{k-1}^l$.

Theorem 5.2.1 — The Rao-Blackwellized particle filter. For the linear part of the state vector, the predicted and posterior densities are

$$p(\mathbf{x}_k^l \,|\, \mathbf{x}_{1:k}^n, \mathbf{z}_{1:k-1}) = \mathcal{N}(\mathbf{x}_k^l \,;\, \hat{\mathbf{x}}_{k|k-1}^l, \mathbf{P}_{k|k-1}^l) \tag{5.57}$$

$$p(\mathbf{x}_k^l \,|\, \mathbf{x}_{1:k}^n, \mathbf{z}_{1:k}) = \mathcal{N}(\mathbf{x}_k^l \,;\, \hat{\mathbf{x}}_k^l, \mathbf{P}_k^l) \tag{5.58}$$

where expectations and covariances are given by

$$[\hat{\mathbf{x}}_k^l, \mathbf{P}_k^l, \hat{\mathbf{x}}_{k|k-1}^l, \mathbf{P}_{k|k-1}^l, \mathbf{S}_k^l] = \mathrm{KF}(\hat{\mathbf{x}}_{k-1}^l, \mathbf{P}_{k-1}^l, \mathbf{z}_k - \mathbf{h}_k). \tag{5.59}$$

with $\mathbf{A}_k$, $\mathbf{C}_k$, $\mathbf{Q}^l$ and $\mathbf{R}^l$ playing the role as transition matrix, measurement matrix, process noise matrix and measurement noise matrix, respectively. For the nonlinear part of the state vector, the predicted and posterior densities are given according to

$$p(\mathbf{x}_k^n \,|\, \mathbf{x}_{1:k-1}^n, \mathbf{z}_{1:k-1}) = \mathcal{N}(\mathbf{x}_k^n \,;\, \boldsymbol{f}_k^n, \mathbf{Q}_k^n) \tag{5.60}$$

$$p(\mathbf{z}_k \,|\, \mathbf{x}_{1:k}^n, \mathbf{z}_{1:k-1}) = \mathcal{N}(\mathbf{z}_k \,;\, \mathbf{h}_k + \mathbf{C}_k \,;\, \hat{\mathbf{x}}_{k|k-1}^l, \mathbf{S}_k^l). \tag{5.61}$$

Proof. For the linear part, the key observation to make is that everything is done conditional on the nonlinear trajectory $\mathbf{x}_{1:k}^n$. Thus, both $\mathbf{x}_{1:k-1}^n$ and $\mathbf{x}_k^n$ are treated as if they were known, i.e., deterministic. Under this conditioning, both the process model and the measurement model of the linear part constitute an LTV system (see page 68). The prediction and the posterior are therefore given by a Kalman filter with the appropriate system matrices. Observe that we have to move $\mathbf{h}_k^n$ over to the left-hand side of (5.54) in order to ensure that it is of the same form as a standard linear measurement model.

For the nonlinear part, the prediction formula (5.60) is nothing more than a restatement of the nonlinear process model (5.53). As for the update formula, a quick glance at (5.61) should give a hint that the product identity again will be our friend. The trick is to re-introduce the linear state by means of the total probability theorem:

$$\begin{aligned} p(\mathbf{z}_k \,|\, \mathbf{x}_{1:k}, \mathbf{z}_{1:k-1}) &= \int p(\mathbf{z}_k \,|\, \mathbf{x}_k^l, \mathbf{x}_k^n) p(\mathbf{x}_k^l \,|\, \mathbf{x}_{1:k}^n, \mathbf{z}_{1:k-1}) \, \mathrm{d}\mathbf{x}_k^l \\ &= \int \mathcal{N}(\mathbf{z}_k - \mathbf{h}_k \,;\, \mathbf{C}_k \mathbf{x}_k^l, \mathbf{R}) \mathcal{N}(\mathbf{x}_k^l \,;\, \hat{\mathbf{x}}_{k|k-1}^l, \mathbf{P}_{k|k-1}^l) \mathrm{d}\mathbf{x}_k^l \\ &= \mathcal{N}(\mathbf{z}_k - \mathbf{h}_k \,;\, \mathbf{C}_k \hat{\mathbf{x}}_{k|k-1}^l, \mathbf{C}_k \mathbf{P}_{k|k-1}^l \mathbf{C}_k^\mathsf{T} + \mathbf{R}). \end{aligned} \tag{5.62}$$

The covariance matrix in this Gaussian is nothing else than the matrix $\mathbf{S}_k^l$ that comes out of the Kalman filtering cycle (5.59). To finalize the proof, we see that (5.61) follows by moving $\mathbf{h}_k$ from the random variable slot to the expectation slot. ■

The densities involved in Theorem 5.2.1 provide everything we need to know to implement an RBPF. Conditional on any particle, i.e., realization of the trajectory $\mathbf{x}_{1:k-1}^n$ we use a Kalman filter to estimate $\mathbf{x}_k^l$. Particles can be sampled using the same tools that were available for non-Rao-Blackwellized particle filters. To obtain the posterior particle weights we use the likelihood (5.61) in the same way as we would do for a non-Rao-Blackwellized particle filter.

5.3 References and chapter remarks

We have only scratched the surface of nonlinear filtering in this chapter. This topic will be covered in more depth in the PhD-level course TK8102 Nonlinear State Estimation. Concerning the EKF and linearization-based methods there are at least three possible directions that the reader should be aware of. First, there is the possibility of implementing iterated EKFs, or solving an EKF-style estimation problem through Gauss-Newton optimization. The relationship between these two viewpoints is a central topic in [42]. Second, there is the possibility of calculating more correct covariances in EKF-style problems. This can in some instances be achieved by deterministic sample-based techniques such as the Unscented Kalman filter [49] or by the so-called Laplace approximation [15]. Third, a general framework for linearization possibilities, that include both the EKF, the UKF and several other possibilities, has recently been published in [38].

Regarding sample-based filtering methods, the most important alternative to the particle filter is the point mass filter. This is just an evaluation of the pdfs involved in the Bayes recursion (4.4) and (4.5) on a discrete grid. While straightforward in one dimension, it becomes intractable in dimensions ≥ 4. For treatment of the point mass filter the reader is referred to [42], and also to PhD theses such as [8] and [1]. Other grid-based techniques include variations of dynamic programming [7] and formulation of state estimation as the solution of a stochastic partial differential equation [60] [24].

The standard textbook on nonlinear filtering by means of particle filters is [73]. Particle filtering has been thoroughly treated in a series of PhD theses at Linköping University. In particular, our treatment of the RBPF is based on Schön's thesis [77]. An alternative version of the RBPF has also been proposed in Hendeby's thesis [43]. This version treats the RBPF as a filter bank, consisting of several linear filters running in parallel. We shall not pursue this line of thought any further here. Instead, the next chapter will focus on the standard approach to filter banks used in target tracking.

There exist several other nonlinear filtering techniques that are reminiscent of particle filtering. In particular, we mention the Ensemble Kalman filter (ENKF) [50] which is used for very high-dimensional problems in e.g. meteorology, and the Daum-Huang filter [25] which attempts to implement Bayes' rule by *moving* the particles instead of adjusting their weigths as done in a conventional particle filter. During the presentation of that paper at the SPIE conference, which the author attended, Fred Daum pointed out that this kind of filter was specifically designed to solve "the problem of high SNR", i.e., low process noise.

The CT model is extensively studied in Chapter 4 and 11 of [3]. Our version of the CT model in (5.15) - (5.17) is slightly different than the "canonical" CT model presented in [3]. This is partly due to a different choice of the order of elements in $\mathbf{x}$. More importantly, we have here opted to develop the CT model from a continuous-time perspective, while a pure discrete-time perspective was used in [3], leading to a different $\mathbf{Q}$ matrix. A comparison of EKF and particle filters for the CT model was published in [30].

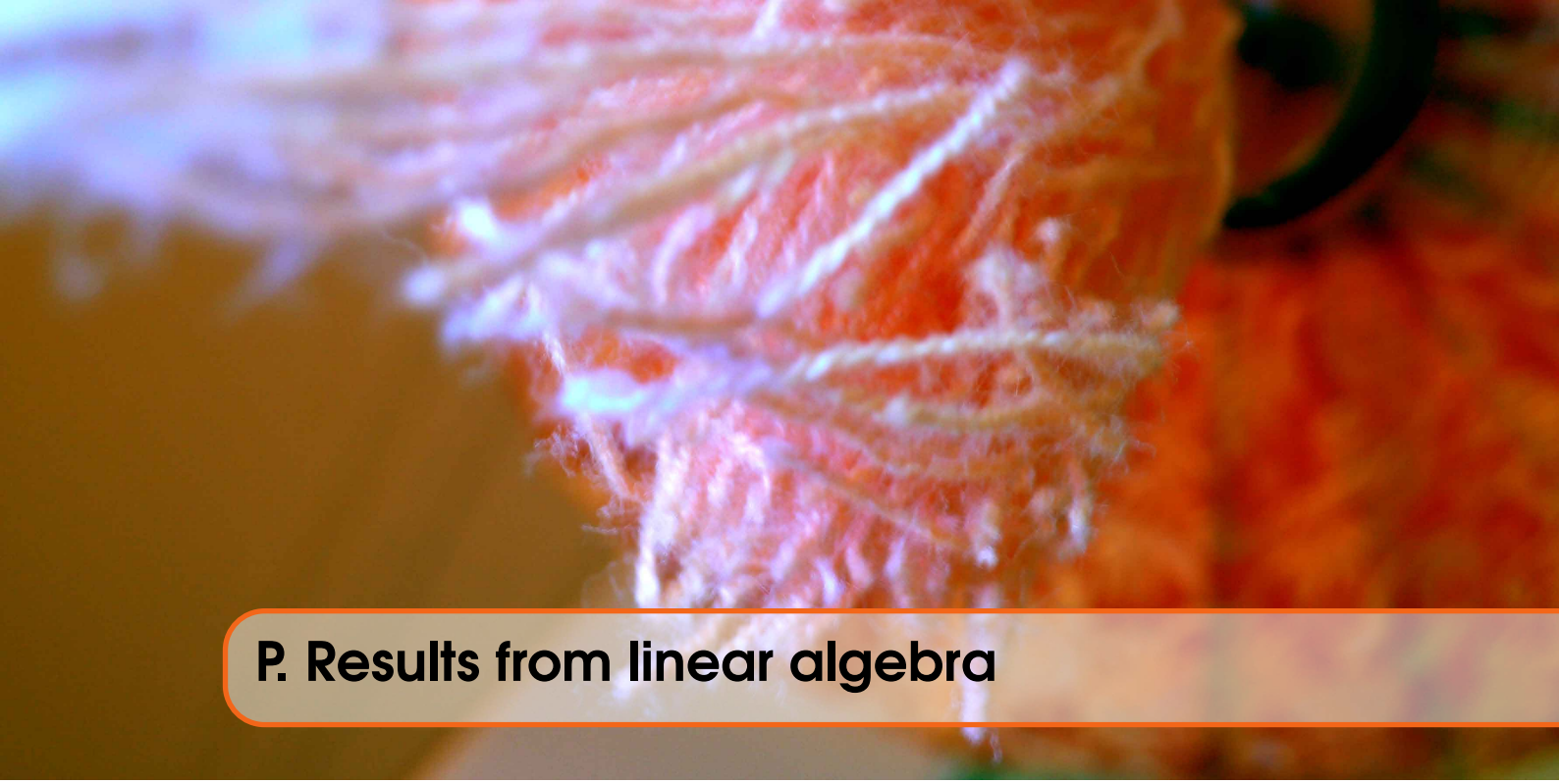
5.4 Exercises

Exercise 5.1 In Example 5.1, we notice that the CT model gets a larger positional covariance than the CV model even though it uses exactly the same values of σ_a and σ_z. Explain where the the additional uncertainty comes from, and how it is channeled into the position covariance. ■

Exercise 5.2 On page 81 it was claimed that importance sampling for the integral in (5.28) also can be used when the proportionality constant in the pdf $\pi(\mathbf{x})$ is unknown. Show that this works when the normalized importance weights in (5.29) are used. ■

Exercise 5.3 Derive the formulas for optimal proposal density with linear measurement model and Gaussian noise. ■

Exercise 5.4 It was claimed on page 88 that in the case of a linear measurement model the optimal proposal density can be implemented using the closed-form expressions in (5.44) and (5.45). Prove this result. **Hint:** Use the product identity. ■



P. Results from linear algebra

P.1 The Schur complement and Boltz' inversion rule

Let the matrix $\mathbf{M}$ be given by

$$\mathbf{M} = \begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix}. \tag{P.1}$$

The Schur complement of the block $\mathbf{D}$ in $\mathbf{M}$ is $\mathbf{A} - \mathbf{B}\mathbf{D}^{-1}\mathbf{C}$, and the Schur complement of the block $\mathbf{A}$ in $\mathbf{M}$ is $\mathbf{D} - \mathbf{C}\mathbf{A}^{-1}\mathbf{B}$. The Schur complement plays a central role when we invert matrices of the form (P.1). With the purpose of obtaining a general formula for inverting such matrices, let us also introduce the matrices

$$\mathbf{L} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ -\mathbf{D}^{-1}\mathbf{C} & \mathbf{I} \end{bmatrix}, \quad \mathbf{U} = \begin{bmatrix} \mathbf{I} & -\mathbf{B}\mathbf{D}^{-1} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \quad \text{and} \quad \mathbf{N} = \begin{bmatrix} \mathbf{A} - \mathbf{B}\mathbf{D}^{-1}\mathbf{C} & \mathbf{0} \\ \mathbf{0} & \mathbf{D} \end{bmatrix}. \tag{P.2}$$

Obviously, the goal here is to replace the inversion of a complicated matrix $\mathbf{M}$ with inversion of a simpler block-diagonal matrix $\mathbf{N}$ which involves the Schur complement.

Theorem P.1.1 — Blockwise matrix inversion. With $\mathbf{M}$, $\mathbf{L}$, $\mathbf{U}$ and $\mathbf{N}$ defined in (P.1) and (P.2), we have the identity $\mathbf{M}^{-1} = \mathbf{L}\mathbf{N}^{-1}\mathbf{U}$.

Proof. By reverse engineering of the desired result, we find that $\mathbf{L}^{-1}\mathbf{M}^{-1} = \mathbf{N}^{-1}\mathbf{U}$ should hold, and furthermore that $\mathbf{M}\mathbf{L} = \mathbf{U}^{-1}\mathbf{N}$ should hold. The left-hand-side of this identity becomes

$$\mathbf{M}\mathbf{L} = \begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ -\mathbf{D}^{-1}\mathbf{C} & \mathbf{I} \end{bmatrix} = \begin{bmatrix} \mathbf{A} - \mathbf{B}\mathbf{D}^{-1}\mathbf{C} & \mathbf{B} \\ \mathbf{0} & \mathbf{D} \end{bmatrix}.$$

For the right-hand-side, we need to invert $\mathbf{U}$. We leave it as an exercise to the reader to show that

$$\mathbf{U}^{-1} = \begin{bmatrix} \mathbf{I} & \mathbf{B}\mathbf{D}^{-1} \\ \mathbf{0} & \mathbf{I} \end{bmatrix}.$$

Thus, the right-hand-side becomes

$$\mathbf{U}^{-1}\mathbf{N} = \begin{bmatrix} \mathbf{I} & \mathbf{B}\mathbf{D}^{-1} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{A}-\mathbf{B}\mathbf{D}^{-1}\mathbf{C} & \mathbf{0} \\ \mathbf{0} & \mathbf{D} \end{bmatrix} = \begin{bmatrix} \mathbf{A}-\mathbf{B}\mathbf{D}^{-1}\mathbf{C} & \mathbf{B} \\ \mathbf{0} & \mathbf{D} \end{bmatrix}.$$

This concludes the proof. ■

Having established this important result, we can express the inverse of a block matrix in two ways which both are useful:

$$\begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix}^{-1} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ -\mathbf{D}^{-1}\mathbf{C} & \mathbf{I} \end{bmatrix} \begin{bmatrix} (\mathbf{A}-\mathbf{B}\mathbf{D}^{-1}\mathbf{C})^{-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{D}^{-1} \end{bmatrix} \begin{bmatrix} \mathbf{I} & -\mathbf{B}\mathbf{D}^{-1} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \tag{P.3}$$

$$= \begin{bmatrix} (\mathbf{A}-\mathbf{B}\mathbf{D}^{-1}\mathbf{C})^{-1} & -(\mathbf{A}-\mathbf{B}\mathbf{D}^{-1}\mathbf{C})^{-1}\mathbf{B}\mathbf{D}^{-1} \\ -\mathbf{D}^{-1}\mathbf{C}(\mathbf{A}-\mathbf{B}\mathbf{D}^{-1}\mathbf{C})^{-1} & \mathbf{D}^{-1}+\mathbf{D}^{-1}\mathbf{C}(\mathbf{A}-\mathbf{B}\mathbf{D}^{-1}\mathbf{C})^{-1}\mathbf{B}\mathbf{D}^{-1} \end{bmatrix} \tag{P.4}$$

The first of these two equations (P.3) is known as Aitken's block diagonalization formula, while the last equation (P.4) is known as Boltz' rule for matrix inversion.

P.2 The matrix inversion lemma

In Boltz' rule (P.4) it can be noted that $\mathbf{A}$ and $\mathbf{D}$ are treated somewhat differently. This is because we chose to express $\mathbf{N}$ in terms of the Schur complement of $\mathbf{D}$. If we instead choose to work with the Schur complement of $\mathbf{A}$, symmetry dictates that we must arrive at

$$\begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix}^{-1} = \begin{bmatrix} \mathbf{A}^{-1}+\mathbf{A}^{-1}\mathbf{B}(\mathbf{D}-\mathbf{C}\mathbf{A}^{-1}\mathbf{B})^{-1}\mathbf{C}\mathbf{A}^{-1} & -\mathbf{A}^{-1}\mathbf{B}(\mathbf{D}-\mathbf{C}\mathbf{A}^{-1}\mathbf{B})^{-1} \\ -(\mathbf{D}-\mathbf{C}\mathbf{A}^{-1}\mathbf{B})^{-1}\mathbf{C}\mathbf{A}^{-1} & (\mathbf{D}-\mathbf{C}\mathbf{A}^{-1}\mathbf{B})^{-1} \end{bmatrix}. \tag{P.5}$$

The matrix inversion lemma, also known as the Woodbury identity, follows from equating the upper diagonal blocks in (P.4) and (P.5):

$$(\mathbf{A}-\mathbf{B}\mathbf{D}^{-1}\mathbf{C})^{-1} = \mathbf{A}^{-1}+\mathbf{A}^{-1}\mathbf{B}(\mathbf{D}-\mathbf{C}\mathbf{A}^{-1}\mathbf{B})^{-1}\mathbf{C}\mathbf{A}^{-1}. \tag{P.6}$$

The matrix inversion lemma is often invoked in derivations of the Kalman filter.

Q. Matrix and vector derivatives

Bibliography

Books

[2] Y. Bar-Shalom and X. R. Li, *Multitarget-Multisensor Tracking: Principles and Techniques*. Storrs, CT, USA: YBS Publishing, 1995 (cited on pages 14, 65, 105, 107, 109, 117, 119, 132).

[3] Y. Bar-Shalom, X. R. Li, and T. Kirubarajan, *Estimation with Application to Tracking and Navigation*. Wiley, 2001 (cited on pages 13, 34, 46, 60, 63, 69, 91, 96, 98, 107).

[5] Y. Bar-Shalom, P. K. Willett, and X. Tian, *Tracking and Data Fusion: A Handbook of Algorithms*. Storrs, CT, USA: YBS Publishing, 2011 (cited on pages 14, 132).

[6] T. D. Barfoot, *State estimation for robotics*. Cambridge University Press, 2018 (cited on page 14).

[9] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer, 2006 (cited on page 46).

[10] S. Blackman and R. Popoli, *Design and Analysis of Modern Tracking Systems*. Norwood, MA, USA: Artech House, 1999 (cited on pages 14, 150).

[20] S. Challa, M. R. Morelande, D. Musicki, and R. J. Evans, *Fundamentals of object tracking*. Cambridge University Press, 2011 (cited on pages 14, 117).

[28] O. Egeland and T. Gravdahl, *Modeling and Simulation for Automatic Control*. Trondheim, Norway: Marine Cybernetics, 2002 (cited on page 174).

[31] G. B. Folland, *Real Analysis: Modern Techniques and Their Applications*. Wiley, 1999 (cited on page 34).

[34] T. I. Fossen, *Handbook of Marine Craft Hydrodynamics and Motion Control*. Wiley, 2011 (cited on page 183).

[39] A. Gelb, *Applied Optimal Estimation*. MIT Press, 1973 (cited on page 69).

[40] P. D. Groves, *Principles of GNSS, Inertial, and Multisensor Integrated Navigation Systems*, 2nd edition. Artech House, 2013 (cited on pages 14, 178).

[42] F. Gustafsson, *Statistical Sensor Fusion*. Lund, Sweden: Studentlitteratur, 2010 (cited on pages 13, 14, 46, 91).

[48] R. A. Johnson and D. W. Wichern, *Applied Multivariate Statistical Analysis*, 5th edition. Prentice Hall, 2002 (cited on page 46).

[51] S. M. Kay, *Fundamentals of Statistical Signal Processing Volume I Estimation Theory*. Prentice-Hall, 1993 (cited on page 14).

[59] R. Mahler, *Statistical Multisource-Multitarget Information Fusion*. Norwood, MA, USA: Artech House, 2007 (cited on pages 14, 46, 110, 153, 156, 159, 165, 166).

[63] K. P. Murphy, *Machine Learning - A Probabilistic Perspective*. Cambridge, MA, USA: MIT Press, 2012 (cited on page 46).

[70] A. Papoulis and S. U. Pillai, *Probability, Random Variables and Stochastic Processes*. McGraw-Hill, 2002 (cited on pages 14, 27, 34, 55, 69).

[73] B. Ristic, S. Arulampalam, and N. Gordon, *Beyond the Kalman Filter: Particle Filters for Tracking Applications*. Artech House, 2004 (cited on pages 68, 89, 91).

[76] S. Särkkä, *Bayesian Filtering and Smooting*. Cambridge University Press, 2013 (cited on page 68).

[84] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. MIT Press, 2006 (cited on page 14).

[85] D. Titterton and J. Weston, *Strapdown Inertial Navigation Technology*, 2nd edition, series Radar, Sonar and Navigation. IET, 2004 (cited on page 14).

[89] B. Vik, *Integrated Satellite and Inertial Navigation Systems*. Department of Engineering Cybernetics, NTNU, 2014 (cited on pages 14, 176).

[91] R. E. Walpole, R. H. Myers, and S. L. Myers, *Probability and Statistics for Engineers and Scientists*. Prentice Hall, 1998 (cited on page 34).

Articles

[1] K. B. Anonsen, "Advances in terrain aided navigation for underwater vehicles", PhD thesis, NTNU, 2010 (cited on page 91).

[4] Y. Bar-Shalom, S. S. Blackman, and R. J. Fitzgerald, "Dimensionless score function for multiple hypothesis tracking", *IEEE Transactions on Aerospace and Electronic Systems*, volume 43, number 1, pages 392–400, Jan. 2007. DOI: 10.1109/TAES.2007.357141 (cited on pages 149, 150).

[7] Y. Barniv, "Dynamic Programming Algorithm for Detecting Dim Moving Targets", in *Multitarget-Multisensor Tracking: Advanced Applications*, Y. Bar-Shalom, Ed., Artech House, 1990, ch. 4, pages 85–155 (cited on page 91).

[8] N. Bergman, "Recursive bayesian estimation", PhD thesis, Linköping University, 1999 (cited on page 91).

[11] J.-L. Blanco, "A tutorial on se(3) transformation parameterizations and on-manifold optimization", MAPIR: Grupo de Percepción y Robótica, Universidad de Málaga, Tech. Rep., Oct. 2014 (cited on page 176).

[12] H. A. P. Blom and E. A. Bloem, "Probabilistic Data Association Avoiding Track Coalescence", *IEEE Transactions on Automatic Control*, volume 45, number 2, pages 247–259, Feb. 2000, ISSN: 0018-9286. DOI: 10.1109/9.839947 (cited on page 130).

[13] E. Brekke and M. Chitre, "The multiple hypothesis tracker derived from finite set statistics", in *20th International Conference on Information Fusion*, Xi'An, China, Jul. 2017, pages 1–8 (cited on pages 136, 163).

[14] E. Brekke, "Clutter Mitigation for Target Tracking", PhD thesis, Norwegian University of Science and Technology (NTNU), Trondheim, Norway, Jun. 2010 (cited on page 66).

[15] E. Brekke and M. Chitre, "A multi-hypothesis solution to data association for the two-frame SLAM problem", *The International Journal of Robotics Research*, volume 34, number 1, pages 43–63, Jan. 2015. DOI: 10.1177/0278364914545674 (cited on pages 91, 191).

[16] E. Brekke, O. Hallingstad, and J. Glattetre, "The signal-to-noise ratio of human divers", in *Proceedings of OCEANS'10*, Sydney, Australia, May 2010 (cited on pages 62, 66, 167).

[17] ——, "Tracking small targets in heavy-tailed clutter using amplitude information", *IEEE Journal of Oceanic Engineering*, volume 35, number 2, pages 314–329, May 2010 (cited on pages 109, 131).

[18] ——, "The Modified Riccati Equation for Amplitude-Aided Target Tracking in Heavy-Tailed Clutter", *IEEE Transactions on Aerospace and Electronic Systems*, volume 47, number 4, pages 2874–2886, Oct. 2011. DOI: 10.1109/TAES.2011.6034670 (cited on page 131).

[19] ——, "Improved target tracking in the presence of wakes", *IEEE Transactions on Aerospace and Electronic Systems*, volume 48, number 2, pages 1005–1017, Apr. 2012 (cited on page 131).

[21] D. Clark, B. Ristic, and B.-N. Vo, "PHD filtering with target amplitude feature", in *Information Fusion, 2008 11th International Conference on*, Cologne, Jun. 2008, pages 1–7 (cited on page 109).

[22] J. L. Crassidis and F. L. Markley, "Attitude estimation using modified rodrigues parameters", in *Proceedings of the Flight Mechanics/Estimation Theory Symposium*, Greenbelt, MD, USA, 1996, pages 71–83 (cited on page 169).

[23] R. Danchick and G. E. Newnam, "Reformulating Reid's MHT method with generalised Murty K-best ranked linear assignment algorithm", *IEE Proceedings - Radar, Sonar and Navigation*, volume 153, pages 13–22, Feb. 2006 (cited on page 125).

[24] F. Daum, "Solution of the zakai equation by separation of variables", *IEEE Transactions on Automatic Control*, volume 32, number 10, pages 941–943, Oct. 1987, ISSN: 0018-9286 (cited on page 91).

[25] F. Daum and J. Huang, "Nonlinear filters with log-homotopy", in *Proc. Signal and Data Processing of Small Targets (SPIE)*, San Diego, CA, USA, Sep. 2007 (cited on page 91).

[26] S. Davey, M. G. Rutten, and B. Cheung, "A Comparison of Detection Performance for Several Track-before-Detect Algorithms", *EURASIP Journal on Advances in Signal Processing*, volume 2008, 2008. DOI: 10.1155/2008/428036 (cited on page 150).

[27] S. Deb, M. Yeddanapudi, K. Pattipati, and Y. Bar-Shalom, "A generalized S-D assignment algorithm for multisensor-multitarget state estimation", *IEEE Transactions on Aerospace and Electronic Systems*, volume 33, number 2, pages 523–538, Apr. 1997, ISSN: 0018-9251. DOI: 10.1109/7.575891 (cited on pages 146–148).

[30] A. L. Flåten and E. Brekke, "Rao-blackwellized particle filter for turn rate estimation", in *Proceedings of IEEE Aerospace Conference*, Big Sky, MT, USA, Mar. 2017 (cited on pages 91, 109).

[32] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "On-manifold preintegration for real-time visual–inertial odometry", *IEEE Transactions on Robotics*, volume 33, number 1, pages 1–21, Feb. 2017, ISSN: 1552-3098. DOI: 10.1109/TRO.2016.2597321 (cited on page 169).

[33] T. E. Fortmann, Y. Bar-Shalom, and M. Scheffe, "Sonar tracking of multiple targets using joint probabilistic data association", *IEEE Journal of Ocean Engineering*, volume 3, pages 173–184, 1983 (cited on pages 119, 132).

[35] B. Gade, M. Kloster, and M. Aronsen, "Non-elliptical validation gate for maritime target tracking", in *2018 21st International Conference on Information Fusion (FUSION)*, Jul. 2018, pages 1301–1308. DOI: 10.23919/ICIF.2018.8455282 (cited on page 117).

[36] K. Gade, "Integrering av treghetsnavigasjon i en autonom undervannsfarkost", FFI, Tech. Rep. 97/03179, 1997 (cited on pages 178, 183).

[37] ——, "Inertial navigation - theory and applications", PhD thesis, NTNU, 2018 (cited on page 183).

[38] Á. F. García-Fernández, L. Svensson, M. R. Morelande, and S. Särkkä, "Posterior linearization filter: Principles and implementation using sigma points", *IEEE Transactions on Signal Processing*, volume 63, number 20, pages 5561–5573, Oct. 2015, ISSN: 1053-587X. DOI: 10.1109/TSP.2015.2454485 (cited on page 91).

[41] M. Guignard, "Lagrangean relaxation", *Top*, volume 11, number 2, 2003 (cited on page 146).

[43] G. Hendeby, "Performance and implementation aspects of nonlinear filtering", PhD thesis, Linköping University, 2008 (cited on page 91).

[44] P. Horridge and S. Maskell, "Real-time tracking of hundreds of targets with efficient exact JPDAF implementation", in *9th International Conference on Information Fusion*, Florence, Jul. 2006 (cited on page 125).

[46] G. P. Huang, A. I. Mourikis, and S. I. Roumeliotis, "Observability-based rules for designing consistent EKF SLAM estimators", *The international Journal of Robotics Research*, volume 29, number 5, pages 502–528, Apr. 2010, ISSN: 02783649. DOI: 10.1177/0278364909353640 (cited on pages 193, 194).

[49] S. J. Julier and J. K. Uhlmann, "Unscented filtering and nonlinear estimation", *Proceedings of the IEEE*, volume 92, number 3, pages 401–422, Mar. 2004, ISSN: 0018-9219. DOI: 10.1109/JPROC.2003.823141 (cited on page 91).

[50] M. Katzfuss, J. R. Stroud, and C. K. Wikle, "Understanding the ensemble kalman filter", *The American Statistician*, volume 70, number 4, pages 350–357, 2016 (cited on page 91).

[52] A. Kong, J. S. Liu, and W. H. Wong, "Sequential imputations and bayesian missing data problems", *Journal of the American Statistical Association*, volume 89, number 425, 1994 (cited on page 83).

[53] D. Lerro and Y. Bar-Shalom, "Interacting Multiple Model Tracking with Target Amplitude Feature", *IEEE Transactions on Aerospace and Electronic Systems*, volume 29, number 2, pages 494–509, Apr. 1993 (cited on page 109).

[54] ——, "Tracking with debiased consistent converted measurements versus EKF", *IEEE Transactions on Aerospace and Electronic Systems*, volume 29, number 3, pages 1015–1022, Jul. 1993, ISSN: 0018-9251 (cited on page 65).

[55] X. R. Li, "Tracking in clutter with strongest neighbor measurements. i. theoretical analysis", *IEEE Transactions on Automatic Control*, volume 43, number 11, pages 1560–1578, Nov. 1998 (cited on page 103).

[57] E. Liland, "AIS aided multi hypothesis tracker", Master's thesis, NTNU, Jun. 2017 (cited on page 145).

[58] M. Longbin, S. Xiaoquan, Z. Yiyu, S. Z. Kang, and Y. Bar-Shalom, "Unbiased Converted Measurements for Tracking", *IEEE Transactions on Aerospace and Electronic Systems*, volume 34, number 3, pages 1023–1027, 1998 (cited on page 65).

[60] M. McDonald and B. Balaji, "Continuous-discrete filtering for dim manoeuvring maritime targets", in *Proceedings of the 10th International Conference on Information Fusion*, Quebec, Canada, Jul. 2007, pages 777–782 (cited on page 91).

[61] M. Montemerlo, "FastSLAM: A factored solution to the simultaneous localization and mapping problem with unknown data association", PhD thesis, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, USA, Jun. 2003 (cited on page 195).

[62] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, "FastSLAM: A factored solution to the simultaneous localization and mapping problem", in *Proceedings of AAAI-02*, Edmonton, AL, Canada, 2002, pages 593–598 (cited on page 195).

[64] D. Musicki, R. Evans, and S. Stankovic, "Integrated Probabilistic Data Association", *IEEE Transactions on Automatic Control*, volume 39, number 6, pages 1237–1241, 1994 (cited on pages 110, 111, 115, 117).

[65] D. Musicki and S. Suvorova, "Tracking in clutter using IMM-IPDA-based algorithms", volume 44, number 1, pages 111–126, Jan. 2008, ISSN: 00189251 (cited on page 132).

[66] J. Neira and J. D. Tardós, "Data association in stochastic mapping using the joint compatibility test", *IEEE Transactions on Robotics and Automation*, volume 17, number 6, pages 890–897, Dec. 2001, ISSN: 1042296X. DOI: 10.1109/70.976019 (cited on page 191).

[67] P. C. Niedfeldt, K. Ingersoll, and R. W. Beard, "Comparison and analysis of Recursive-RANSAC for multiple target tracking", *IEEE Transactions on Aerospace and Electronic Systems*, volume 53, number 1, pages 461–476, Feb. 2017, ISSN: 0018-9251 (cited on pages 131, 132, 149).

[68] P. C. Niedfeldt, "Recursive-ransac: A novel algorithm for tracking multiple targets in clutte", PhD thesis, Brigham Young University, 2014 (cited on page 149).

[69] J. Olofsson, E. Brekke, T. I. Fossen, and T. A. Johansen, "Spatially indexed clustering for scalable tracking of remotely sensed drift ice", in *Proceedings of IEEE Aerospace Conference*, Big Sky, MT, USA, Mar. 2017 (cited on page 125).

[72] D. Reid, "An algorithm for tracking multiple targets", *IEEE Transactions on Automatic Control*, volume 24, number 6, pages 843–854, Dec. 1979 (cited on pages 133, 136).

[74] S. Roumeliotis, G. Sukhatme, and G. A. Bekey, "Circumventing dynamic modeling: Evaluation of the error-state Kalman filter applied to mobile robot localization", in *Proceedings of ICRA*, volume 2, 1999, 1656–1663 vol.2 (cited on page 183).

[75] D. J. Salmond, "Tracking in uncertain environments", Royal Aerospace Establishment, UK, Tech. Rep. AW 121, Sep. 1989 (cited on page 46).

[77] T. Schön, "On computational methods for nonlinear estimation", PhD thesis, Linköping University, 2003 (cited on page 91).

[79] F. Schweppe, "System identification", in *Uncertain Dynamic Systems*, Prentice Hall, 1979, ch. 14 (cited on page 69).

[80] R. Smith, M. Self, and P. Cheeseman, "Estimating uncertain spatial relationships in robotics", *Autonomous robot vehicles*, pages 167–193, 1990 (cited on page 186).

[82] E. F. Wilthil, A. L. Flaten, and E. F. Brekke, "A target tracking system for asv collision avoidance based on the pdaf", English, in *Sensing and Control for Autonomous Vehicles*, P. Fossen and Nijmeijer, Eds., volume 474, Alesund, Norway: Springer, 2017, pages 269–288 (cited on pages 63, 64, 66).

[83] T. Tengesdal, "Uncertainty management in a scenario-based mpc for collision avoidance", Master's thesis, NTNU, Jun. 2019 (cited on page 80).

[86] L.-C. N. Tokle, "Multi target tracking using random finite sets with a hybrid state space and approximations", Master's thesis, NTNU, Sep. 2018 (cited on page 46).

[87] N. Trawny and S. I. Roumeliotis, "Indirect kalman filter for 3d attitude estimation - a tutorial for quaternion algebra", MARS-LAB, Tech. Rep., 2005 (cited on page 183).

[88] C. Van Loan, "Computing integrals involving the matrix exponential", *IEEE Transactions on Automatic Control*, volume 23, number 3, pages 395–404, 1978 (cited on page 61).

[90] B.-N. Vo, S. Singh, and A. Doucet, "Sequential Monte Carlo methods for multitarget filtering with random finite sets", *IEEE Transactions on Aerospace and Electronic Systems*, volume 41, number 4, pages 1224–1245, Oct. 2005, ISSN: 0018-9251. DOI: 10.1109/TAES.2005.1561884 (cited on page 159).

[92] P. Willett, Y. Ruan, and R. Streit, "PMHT: Problems and Some Solutions", *IEEE Transactions on Aerospace and Electronic Systems*, volume 38, number 3, pages 738–754, Jul. 2002, ISSN: 0018-9251. DOI: 10.1109/TAES.2002.1039396 (cited on page 150).

[93] J. Williams, "An efficient, variational approximation of the best fitting multi-Bernoulli filter", *IEEE Transactions on Signal Processing*, volume 63, number 1, pages 258–273, Jan. 2015, ISSN: 1053-587X. DOI: 10.1109/TSP.2014.2370946 (cited on page 130).

[94] ——, "Marginal multi-Bernoulli filters: RFS derivation of MHT, JIPDA, and association-based MeMBer", *IEEE Transactions on Aerospace and Electronic Systems*, volume 51, number 3, pages 1664–1687, Jul. 2015 (cited on pages 117, 162).

[95] J. Williams and R. Lau, "Approximate evaluation of marginal association probabilities with belief propagation", *IEEE Transactions on Aerospace and Electronic Systems*, volume 50, number 4, pages 2942–2959, Oct. 2014 (cited on page 125).

[96] E. F. Wilthil, E. Brekke, and O. B. Asplin, "Track initiation for maritime radar tracking with and without prior information", in *Proc. Fusion*, Cambridge, UK, Jul. 2018 (cited on page 114).

[97] Y. Xia, K. Granström, L. Svensson, and Á. F. García-Fernández, "An implementation of the Poisson Multi-Bernoulli mixture trajectory filter via dual decomposition", in *2018 21st International Conference on Information Fusion (FUSION)*, Jul. 2018, pages 1–8. DOI: 10.23919/ICIF.2018.8455236 (cited on page 146).

Sources and resources from the web

[29] M. A. T. Figueiredo, "Lecture notes on Bayesian estimation and classification", Instituto de Telecomunicacoes, Portugal., Oct. 2004 (cited on page 34).

[45] G. B. Huang, "Conditional and marginal distributions of a multivariate Gaussian", Accessed 17th of June 2017, Feb. 2010, [Online]. Available: https://gbhqed.wordpress.com/2010/02/21/conditional-and-marginal-distributions-of-a-multivariate-gaussian (cited on pages 46, 196).

[47] T. A. Johansen and T. I. Fossen, "The eXogenous Kalman filter (XKF)", Submitted to International Journal of Control, 2015 (cited on page 194).

[56] E. Liland, “An ILP approach to multi hypothesis tracking”, Specialization project at NTNU, Dec. 2016 (cited on page 146).

[71] J. Pedersen, “Surveillance of the channel”, Specialization Project at NTNU, Dec. 2017 (cited on page 130).

[78] T. B. Schön and F. Lindsten, “Manipulating the Multivariate Gaussian Density”, Accessed 17th of June 2017, Jan. 2011, [Online]. Available: `user.it.uu.se/~thosc112/pubpdf/schonl2011.pdf` (cited on page 46).

[81] J. Solà, “Quaternion kinematics for the error-state KF”, Mar. 2015, [Online]. Available: `http://www.iri.upc.edu/people/jsola/JoanSola/objectes/notes/kinematics.pdf` (cited on pages 14, 171, 174, 178–180, 183).