Linear dynamical models, Kalman filtering and statistics.
Lecture notes to IN-ST 259

Erik Bølviken Nils Christophersen Geir Storvik

University of Oslo, October 1998

Preface

This course addresses students in both computer science (mathematical modeling) and statistics at the advanced undergraduate level. The background comprises a knowledge of linear deterministic dynamical systems (as given in MA-IN 126), linear algebra (MA 104), and the basic concepts of probability and statistics (ST 101). In MA-IN 126, linear dynamical systems were studied in continuous time using differential equations but here we shall work exclusively with *discrete* linear systems represented by difference equations. This has several advantages:

- When random variables are introduced to turn deterministic models into their stochastic counterparts, the resulting theory is much simpler in the discrete case than in the continuous one.
- The use of digital computers naturally leads to discrete mathematical formulations and the collection of observations in discrete time.
- Some dynamical systems, for example in economics, are defined only at discrete points in time. In any case, discrete time models may approximate continuous ones.

In the first part of these notes, we show how continuous linear, deterministic systems may be "discretized", i.e. how difference equations may be derived from the continuous formulations. The emphasis is on state space models and the important system theoretic concepts of observability and reachability are defined.

After describing what may be termed the deterministic "skeletons" in our context, stochastic models are introduced in the second and major part of these notes by adding "noise" (i.e. random variables) to the deterministic part. The Gaussian distribution in multidimensional space plays an important role when analyzing such models. A brief introduction to the theory of Gaussian random variables and processes is therefore included.

After the discussion of the Gaussian distribution, an account of linear stochastic state space models and the Kalman filter is given. This topic is also treated in numerous other books and articles in the literature. Most of these accounts have been written in engineering contexts, although there has been a steadily growing interest in statistics, and many books on so-called time series analysis now include an exposition of Kalman filtering. In deciding to write out these notes in spite of the large body of written texts, we have been motivated by our need for addressing students in both mathematical modeling and statistics at the advanced undergraduate level. This suggests a text combining the main aspects of conventional Kalman filtering with an exposition of the unifying role these techniques can play within statistics. Apart from the elegant formulation of linear, dynamical statistical models obtained thereby, the applied statistician gains enormous modeling flexibility by mastering this tool. He or she may, for example, easily include serially correlated errors in a regression, fit all sorts of unbalanced models for analysis of variance, allow parameters to fluctuate dynamically in time, or work with Bayesian versions of standard linear models. Through the Kalman filter he or she has a unified way of fitting such models and to make predictions or forecasts. Other benefits are automatic handling of missing data or, more generally, irregular patterns of observations. Students specializing in mathematical modeling will, on their part, in a work situation often find themselves confronted with problems belonging to statistics. It is undoubtedly worthwhile to teach them how to use the Kalman filter apparatus to fit statistical models such as regression or dynamic regression. Adding stochastic simulation to assess uncertainty (and even to test statistical hypotheses should the need arise), mathematical modelers, mastering, say *Matlab* , should, with some training, be able to do work often thought to belong to the realm of specialist statisticians. With the present course, we attempt to a take a step in this direction.

To illustrate concepts and techniques beyond the simplest conventional textbook examples, one needs computer simulations. Modern software, such as *Matlab* , enables the students to get a deeper understanding of the main ideas and it becomes feasible to work with realistic examples. *Matlab* will be an integral part of the course and many exercises involve experimentation in this powerful software. In many cases the same examples are used in subsequent exercises. We therefore recommend the student to write their *Matlab* code as functions `m`-files, making it easy to utilize and modify in subsequent exercises. A separate appendix listing the most important commands and functions is included in order to make the *Matlab* implementations easier to work out.

We strongly discourage using the software simply as a black box. As when using a calculator, one must be able to check that the results are reasonable. (Students having IN 227 may have an advantage here, as this course provides a good general background in numerical linear algebra.)

The present notes, prepared for the autumn of 1998 is an extended version of the one from autumn 1997. A chapter discussing non-linear models is included. Further, many errors and misprints have been corrected. We are grateful to collegues and students for all the comments we have recived. We will in particular thank Bent Natvig for

careful proof-reading on most of the chapters and many valuable comments. Further, we will thank Anders Løland for finding a huge number of misprints!

Contents

Chapter 1

Introduction

Linear models of dynamical systems exist in various forms and may be categorized in different ways. In this course, we will separate between continuous and discrete representations and either of these may again be deterministic or stochastic. Dynamic linear models of these types are applied in many different fields. From a course in differential equations, we know that a diverse set of dynamical phenomena are well described by linear deterministic differential equations. Such phenomena include voltages and currents in electrical circuits, motions governed by Newton's laws, and biological systems that can be divided into linear compartments. By introducing random variables into such models, we are able to describe dynamic patterns that fluctuate in an unpredictive way due to noise or errors, or due to a complexity that eludes our ability to describe the systems in deterministic terms.

In general, linear models are widely used to describe dynamical behavior throughout the natural sciences and also in other fields such as economics. However, this does not imply that modeling of a dynamical system is necessarily a simple matter. In some cases, for example in physics, formulation of a model may be rather straightforward because the governing equations are known. But in other cases involving "softer" disciplines, model development may be a tedious process where various approximations must be assessed before one arrives at an adequate model representation.

In this course, we will assume that a model of the system has been derived, and that observations (generally noise-corrupted) of some of the variables or combinations of variables in the system are available. From the model *and* the observations, the task is to estimate variables and parameters that are not directly observable. A huge body of theory has been developed to treat various aspects of such problems, including realization theory, system identification, filtering theory, time series analysis, and different types of statistical estimation theory. Here we can only focus on some of these aspects, and so-called state estimation will play a prominent role. To make the presentation more concrete, we start with an example that will be used repeatedly.

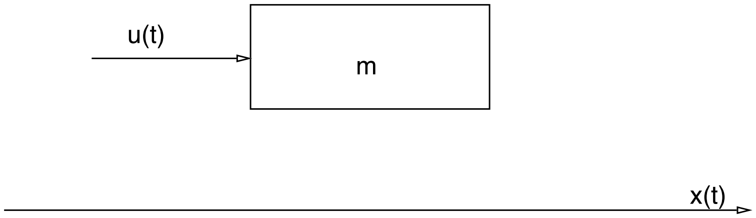

Figure 1.1: A body of mass m moving in one dimension under the influence of an external force $u(t)$

Example 1 (A moving body)
Consider the system in Figure 1.1, where a body of mass m moves in one dimension under the influence of an external force $u(t)$.

From Newton's 2nd law of motion, one obtains a linear second order differential equation for the displacement (position) $x(t)$. (See for example [2].)

$$m\ddot{x}(t) = u(t). \tag{1.1}$$

Given the initial conditions $x(0)$ and $\dot{x}(0)$, the motion of the body is completely determined once the *input* function $u(t)$ for $t \geq 0$ is known. The position and the velocity taken together as a vector will be called the *state vector* because these variables contain all information from the past that is necessary to predict the future behavior of the body. (Technically, $u(t)$ must be at least piecewise continuous to secure a unique solution of the differential equation.) The solution $x(t)$, which is the sum of the homogeneous part (involving the initial conditions) and a particular part (determined by $u(t)$), can only be expressed in closed form for certain input functions. However, it is a simple matter to solve equation (1.1) numerically for an arbitrary $u(t)$. □

Eq. (1.1) is a *model* of the physical system in Figure 1.1. The model is derived from so-called first principles (Newton's laws in this case), and gives an accurate description of the motion provided factors such as air resistance and relativistic effects are negligible. A representative problem of the ones we will consider in this course is the following: Suppose that the force $u(t)$ is the sum of a known deterministic part and an unknown part that can only be described in statistical terms (e.g. wind gusts). Furthermore, the position of the body is measured at discrete times with fairly large errors. Our goal is to estimate the speed of the body as well as improving the raw position data. To do this, we use the model, properly implemented on a computer, in combination with the observations. (If we consider a satellite or a missile instead of our innocent body, such a task is of more obvious interest.) The scheme used is the Kalman filter named after Rudolf E. Kalman who published it in 1960. This is called state estimation. If one also wants to estimate the mass of the body from the same observations, that will be classified as parameter estimation.

Before deriving the Kalman filter, we need to be able to discretize eq. (1.1). This is carried out in Chapter 2, which also contains a brief exposition of the linear system theoretic concepts of observability and reachability. These properties, defined here for deterministic systems, play a role when analyzing the performance of the Kalman filter. Because most of the subsequent theory will rely on stochastic theory, Chapter 3 discuss random variables and process in general and Gaussian processes in particular. In Chapter 4, the stochastic state space model is introduced, and some of its properties are discussed. The derivation of the Kalman filter rely heavily on general theory for estimation of random variables based on a set of estimation. This will be treated in Chapter 5. In Chapter 6 the important features of the Kalman filter are laid out and several examples and additional properties are provided in Chapter 8. Prediction and so-called smoothing (where both past and future observations are used in estimating the state vector) are treated in Chapter 9. When putting up a model, many parameters have to be specified. In practice, several of these will be unknown, but can be estimated from data. This issue will be discussed in Chapter 10. In Chapter 7 it is explained how many seemingly different problems in statistics fall within the Kalman filter framework. Finally, Appendix A contains a list of useful *Matlab* commands as well as the *Matlab* code used to program many of the examples.

Bibliographic notes

As noted, the Kalman filter was published in 1960 [9]. Since then an enormous body of literature has been produced on various aspects of the Kalman filter. (A few years ago, a literature count gave 200,000 references.) The filter was originally designed to give a recursive solution to the classical linear least squares estimation problem in signal processing and control theory. Recent references based on this engineering tradition include [3]; [12]; [6]. The last book contains a historical overview. These books also contain additional material on the linear system theoretic properties of observability and reachability (our Chapter 2), Kalman filter stability, various covariance recursions, modeling errors (our Chapter 8) and smoothing (our Chapter 9).

Over time, the Kalman filter has taken root in many other areas. But it took over 20 years before the filter became part of mainstream statistics, although the filter is readily interpreted as a recursive Bayesian estimator. This illustrates an unfortunate 'cultural gap' which sometimes exists between disciplines. Today, the Kalman filter is treated in full from a statistical viewpoint in books such as [8] and [14]. The latter emphasizes in particular on the Bayesian approach to dynamical systems, see also [7]. Many of the mainstream textbooks in statistical time series also cover Kalman filtering.

Chapter 2

Linear deterministic dynamical models.

We will now study what may be called the deterministic “skeletons” of our dynamic linear models. This simplifies the exposition of discretization as it allows us to draw directly on material from courses in ordinary differential equations. Furthermore, the concepts of observability and reachability are most easily introduced in a deterministic setting.

2.1 Discretization of continuous systems

Instead of considering 2nd order, or more generally nth order, differential equations, it is frequently advantageous to transform the model into a set of coupled first order differential equations, i.e. to the so-called state space form . As noted in the previous chapter, the state vector contains the internal variables needed at a certain point in time to solve the model equations. Consider again our motion of body example:

Example 1 (A moving body, cont.)
For our 2nd order example, the two state variables are $x_1(t) = x(t)$ and $x_2(t) = \dot{x}(t)$. Eq. (1.1) can then be written as:

$$\dot{x}_1(t) = x_2(t), \tag{2.1a}$$
$$\dot{x}_2(t) = u(t)/m. \tag{2.1b}$$

In this form, the equations are readily solved in some numerical software, see Appendix A.

Assume that the position $z(t) = x_1(t)$ is measured (free of errors) at discrete times $t_k, k = 1, 2, \ldots$. The complete model, comprising the dynamical part in eq. (2.1) and the measurement, may be written in vector-matrix form as

$$\begin{pmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} x_1(t) \\ x_2(t) \end{pmatrix} + \begin{pmatrix} 0 \\ 1/m \end{pmatrix} u(t), \tag{2.2a}$$

$$z(t_k) = \begin{pmatrix} 1 & 0 \end{pmatrix} \begin{pmatrix} x_1(t_k) \\ x_2(t_k) \end{pmatrix}, \tag{2.2b}$$

where $\mathbf{x} = \begin{pmatrix} x_1 & x_2 \end{pmatrix}^T$ is the state vector. □

In general, we will consider a linear continuous nth order system given by:

$$\dot{\mathbf{x}}(t) = \mathbf{A}(t)\mathbf{x}(t) + \mathbf{B}(t)\mathbf{u}(t), \qquad \text{(system)} \tag{2.3a}$$

$$\mathbf{z}(t_k) = \mathbf{H}(t_k)\mathbf{x}(t_k), \qquad \text{(observations)} \tag{2.3b}$$

where $\mathbf{x}(t)$, $\mathbf{z}(t)$ and $\mathbf{u}(t)$ (the input function) are vectors having dimensions n, m and p, respectively. The time interval between measurements is denoted **the sampling time** and is not necessarily uniform. Note that the system may be time-varying since the defining matrices depend on t.

Physically, this corresponds to the fact that the body will not come to rest at the origin unless it is released there with zero velocity.

Eq. (2.3) forms a natural starting point for the discretization procedure. Recall the general solution to eq. (2.3a) [2]:

$$\mathbf{x}(t) = \Phi(t, t_0)\mathbf{x}(t_0) + \int_{t_0}^{t} \mathbf{\Phi}(t, s)\mathbf{B}(s)\mathbf{u}(s)ds \tag{2.4}$$

where $\mathbf{\Phi}(t, t_0)$ is the so-called transition matrix. We will denote this solution the superposition integral since the complete solution is a superposition of the homogeneous part, including the initial conditions, and the particular solution including the input function $\mathbf{u}(t)$. In the time-invariant case we have $\mathbf{\Phi}(t, t_0) = e^{\mathbf{A}\cdot(t-t_0)}$ (cf. [2]).

Take $t_0 = t_{k-1}$ and $t = t_k$. Then

$$\mathbf{x}(t_k) = \mathbf{\Phi}(t_k, t_{k-1})\mathbf{x}(t_{k-1}) + \int_{t_{k-1}}^{t_k} \mathbf{\Phi}(t_k, s)\mathbf{B}(s)\mathbf{u}(s)ds. \tag{2.5}$$

If $\mathbf{u}(t)$ is a constant $\mathbf{u}(t_{k-1})$ over the interval $[t_{k-1}, t_k)$, we may write the integral term as $\mathbf{\Psi}(t_k, t_{k-1})\mathbf{u}(t_{k-1})$ where

$$\mathbf{\Psi}(t_k, t_{k-1}) = \int_{t_{k-1}}^{t_k} \mathbf{\Phi}(t_k, s)\mathbf{B}(s)ds. \tag{2.6}$$

If $\mathbf{u}(t)$ is not a constant over the sampling interval, it is still customary to use the same notation although now only the composite term $\mathbf{\Psi}(t_k, t_{k-1})\mathbf{u}(t_{k-1})$ can be obtained from the continuous formulation. Simplifying the notation writing k instead of t_k and using only one time argument in the $\mathbf{\Phi}$ and $\mathbf{\Psi}$ matrices (i.e $\mathbf{\Phi}(t_k, t_{k-1}) = \mathbf{\Phi}(k-1)$), we arrive at the discrete state space model that will be employed throughout the course:

$$\mathbf{x}(k) = \mathbf{\Phi}(k-1)\mathbf{x}(k-1) + \mathbf{\Psi}(k-1)\mathbf{u}(k-1), \tag{2.7a}$$

$$\mathbf{z}(k) = \mathbf{H}(k)\mathbf{x}(k). \tag{2.7b}$$

Here $k = 1, 2, \ldots$. Note that the system equation (2.7a) is a *difference equation.* Numerical software such as *Matlab* have general functions for performing such a transform (see Appendix A).

Example 1 (A moving body, cont.)
For our 2nd order example, the discrete system matrices will be time-invariant, i.e. $\mathbf{\Phi}(k-1) = \mathbf{\Phi}$ and $\mathbf{\Psi}(k-1) = \mathbf{\Psi}$ for all k, and are easily computed symbolically as:

$$\mathbf{\Phi} = \begin{pmatrix} 1 & T \\ 0 & 1 \end{pmatrix}, \qquad \mathbf{\Psi} = \begin{pmatrix} T^2/2m \\ T/m \end{pmatrix},$$

where T is the sampling time. (Can you explain why the elements in these matrices make sense physically?) □

Stability in the discrete case is defined in a similar way as for continuous systems. Obviously, a discretized system is stable if the continuous version is so. In the time-invariant case where $\mathbf{\Phi} = e^{\mathbf{A} \cdot T}$, the eigenvalues λ_i^d $(i = 1, \ldots, n)$ of $\mathbf{\Phi}$ are given by $\lambda_i^d = e^{\lambda_i T}$, where λ_i are the eigenvalues of $\mathbf{A}$. If all λ_i lie in the left half of the complex plane, the eigenvalues of $\mathbf{\Phi}$ will be within the unit circle. This is the stability criterion for discrete systems. (These results are treated in more detail in the Exercise 2.1.)

2.2 Stability, reachability and observability

An important question regarding dynamical systems is their equilibrium points and whether these are stable or not [2]. An equilibrium point $\mathbf{x}_e$ satisfies $\dot{\mathbf{x}}_e(t) = 0$. For the homogeneous part of the differential equation in eq. (2.3a) (discarding the second term on the right hand side), the origin is clearly such a point. The origin is called globally asymptotically stable if $||\mathbf{x}(t)|| \to 0$ as $t \to \infty$ for any $\mathbf{x}(0)$. A linear time-invariant system is stable in this sense if and only if all eigenvalues of $\mathbf{A}$ lies strictly in the left part of the complex plane. The solution then comprises sums of *damped* exponential functions. Considering the effects of input functions $\mathbf{u}(t)$, a system with a globally asymptotically stable *homogeneous part* will have a bounded solution ($||\mathbf{x}(t)||$ remains finite) provided $\mathbf{u}(t)$ is itself bounded [2].

For Example 1, eq. (2.2a), both eigenvalues are equal to 0 and are not strictly in the left half plane. The system is therefore not globally asymptotically stable.

Reachability and observability are linear system theoretic concepts that are unrelated to stability. They lead to conditions that in a sense guarantee the model to be well posed for state estimation and identification. We will only consider time-invariant discrete systems (the sampling time T and the matrices $\mathbf{\Phi}, \mathbf{\Psi}$ and $\mathbf{H}$ are fixed), but the ideas may be extended to more general situations

Definition 2.1
A system is **reachable** *if there exists a finite sequence of input vectors* $\mathbf{u}(k)$, $k = 0, 1, \ldots, k'$ *transferring the system from any initial state* $\mathbf{x}(0)$ *to any final state* $\mathbf{x}(k'+1)$.

Intuitively, this implies that the state vector may be controlled from any point to any other point by selecting a proper sequence of inputs. The necessary and sufficient condition for reachability is derived from eq. (2.7a) by writing out the recursion as follows:

$$\begin{aligned}\mathbf{x}(n) &= \mathbf{\Phi}^n\mathbf{x}(0) + \mathbf{\Phi}^{n-1}\mathbf{\Psi}\mathbf{u}(0) + \cdots + \mathbf{\Psi}\mathbf{u}(n-1) \\ &= \mathbf{\Phi}^n\mathbf{x}(0) + \mathbf{W}_c\mathbf{U}. \end{aligned} \tag{2.8}$$

Here n is the dimension of the state vector and

$$\mathbf{W}_c = \begin{pmatrix} \mathbf{\Psi} & \mathbf{\Phi}\mathbf{\Psi} & \cdots & \mathbf{\Phi}^{n-1}\mathbf{\Psi} \end{pmatrix},$$
$$\mathbf{U} = \begin{pmatrix} \mathbf{u}(n-1) \\ \mathbf{u}(n-2) \\ \vdots \\ \mathbf{u}(0) \end{pmatrix}.$$

We see that any vector $\mathbf{x}(n)$ can be reached if and only if the $n \times pn$ matrix $\mathbf{W}_c$ has n independent columns, i.e. the rank is n. Note that for $p > 1$ there is no unique sequence of input vectors. There is no reason to consider $k' > n-1$, since it can be shown (by the Caley - Hamilton theorem) that the rank of $\mathbf{W}_c$ will be maximal for $k' = n-1$.

Definition 2.2
The system is **observable** *if any initial state* $\mathbf{x}(0)$ *can be determined from* $\mathbf{z}(k), k = 1, \ldots, k''$ *for some finite* k''.

This condition implies that the state vector can be reconstructed from a finite sequence of observations, since by knowing $\mathbf{x}(0)$ (and $\mathbf{u}(k)$), the state vector can be computed for all times. Assume for the time being that all inputs are 0. The necessary and sufficient condition for observability is derived by considering the set of equations:

$$\begin{aligned}
\mathbf{z}(1) &= \mathbf{H}\mathbf{x}(1) = \mathbf{H}\mathbf{\Phi}\mathbf{x}(0) \\
\mathbf{z}(2) &= \mathbf{H}\mathbf{x}(2) = \mathbf{H}\mathbf{\Phi}^2\mathbf{x}(0) \\
&\vdots \\
\mathbf{z}(n) &= \mathbf{H}\mathbf{\Phi}^n\mathbf{x}(0),
\end{aligned}$$

or in a more compact form:

$$\begin{pmatrix} \mathbf{H} \\ \mathbf{H}\mathbf{\Phi} \\ \vdots \\ \mathbf{H}\mathbf{\Phi}^{n-1} \end{pmatrix} \mathbf{\Phi}\mathbf{x}(0) = \begin{pmatrix} \mathbf{z}(0) \\ \mathbf{z}(1) \\ \vdots \\ \mathbf{z}(n-1) \end{pmatrix}.$$

Given that $\mathbf{\Phi} = e^{\mathbf{A}\cdot T}$ is always nonsingular (since the eigenvalues are always non-zero), this set of linear equations in $\mathbf{x}(0)$ has a unique solution if and only if the $nm \times n$ matrix

$$\mathbf{W}_o = \begin{pmatrix} \mathbf{H} \\ \mathbf{H}\mathbf{\Phi} \\ \vdots \\ \mathbf{H}\mathbf{\Phi}^{n-1} \end{pmatrix}$$

has n independent rows, i.e. the rank is n. We might, of course, have used the matrix $\mathbf{W}_o\mathbf{\Phi}$ but this would not have complied with the literature standard. The condition will be the same when known inputs affect the system (check that yourself) and, as with reachability, there is no need to consider the case $k'' > n$.

If the system is not observable, the state vector cannot be determined even from noise - free measurements. It seems reasonable that this may have undesirable effects on a method that attempts to estimate the state vector from *noisy* observations. As we will see in Chapter 8, observability is one of the necessary conditions that must be satisfied in order to secure that the estimation problem is well posed in a certain sense. The other condition turns out to be reachability extended to include stochastic inputs.

2.3 Problems

Exercise 2.1 (Eigenvalues and matrix exponentials)
Matrix exponentials are important for discretization of continuous dynamic systems (that is the transition from differential equations to difference equations). We will in this exercise consider the *eigenvalues* of matrix exponentials.

Consider the following matrices

$$\mathbf{A}_1 = \begin{pmatrix} a & 0 \\ 0 & -a \end{pmatrix}, \qquad \mathbf{A}_2 = \begin{pmatrix} a & 1 \\ 0 & a \end{pmatrix}$$

and the Hilbert matrix $\mathbf{H}_n$ of order n. (The Hilbert matrix is the $n \times n$ matrix with (i, j)'th element $h_{ij} = 1/(i+j-1)$.) This matrix is known as being difficult to handle numerically.

The Hilbert matrix may be computed directly in *Matlab* with the command `hilb(n)`.

(a) What are the eigenvalues of $\mathbf{A}_1$ and $\mathbf{A}_2$?
(You may see this directly from the matrices because of their particular form.)

(b) The matrix exponential for an $n \times n$ matrix $\mathbf{A}$ is given by

$$e^{\mathbf{A}} = \mathbf{I} + \mathbf{A} + \mathbf{A}^2/2! + \cdots$$

Calculate by hand $e^{\mathbf{A}_1}$ and $e^{\mathbf{A}_2}$.

(c) More difficult: Can you prove what we now might think? It looks like the general connection is

$$\lambda(e^{\mathbf{A}})_i = e^{\lambda(\mathbf{A})_i}.$$

(Hint: Similarity-transformations are useful if you remember this from your linear algebra course.)

(d) Check the answer you got in (a) in *Matlab* for $a = 2$. (Use `eig`.)

(e) Check the answer you got in (b) in *Matlab* (use `expm`) for $a = 2$.

(f) Use *Matlab* to check if the results in (c) is correct for $\mathbf{H}_n$ with $n = 2, 3, \ldots$.
Is it correct if you try with the 4×4-matrices

$$\mathbf{A}_3 = \begin{pmatrix} \mathbf{A}_1 & \mathbf{0} \\ \mathbf{0} & \mathbf{A}_2 \end{pmatrix} \qquad \text{and} \mathbf{A}_4 = \begin{pmatrix} \mathbf{A}_1 & \mathbf{1} \\ \mathbf{1} & \mathbf{A}_2 \end{pmatrix}$$

where $\mathbf{0}$ is the 2×2 matrix with only zeros, while $\mathbf{1}$ only contain ones?

(Hint: Look at the commands `ones` and `zeros`.)

Exercise 2.2 (A moving body)

(a) Show that the system in Example 1 (the discretized version) is reachable and explain why this is reasonable physically.

(b) Verify that our example is observable if the position is measured, but not if *only* velocity is measured. Why is this?

Exercise 2.3 (A deterministic physical system)
A pendulum with small angular movements, $\theta(t)$, can be described by the equation

$$L\ddot{\theta}(t) + g\theta(t) = u(t)$$

where L is the rod length, g is the gravitational acceleration and $u(t)$ is the acceleration we can incur.

(a) Put the system in state space form ($x_1(t) = \theta(t)$ and $x_2(t) = \dot{\theta}(t)$) and show that we get:

$$\begin{pmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ -g/L & 0 \end{pmatrix} \begin{pmatrix} x_1(t) \\ x_2(t) \end{pmatrix} + \begin{pmatrix} 0 \\ 1/L \end{pmatrix} u(t).$$

(b) Intuitively, do you think this system is stable? Show that the eigenvalues are pure imaginary.

(c) Discretize the system with $T = 0.5\,\mathrm{s}$, that is, calculate $\mathbf{\Phi} = e^{\mathbf{A}\cdot T}$.

(d) Set $L = 1\,\mathrm{m}$, $g = 10\,\mathrm{m/s}^2$, $\theta(0) = 0.1$, $\dot{\theta}(0) = 0$ and $u(t) = 0$, $t > 0$. Solve the differential equations in *Matlab* by use of the routine `ode45` and make plots of $\theta(t)$ and $\dot{\theta}(t)$.

(You should try to write a so-called `m`-file which is a good training-exercise.)

(e) Solve the same system again, but now with the routine `lsim`. Again make plots of $\theta(t)$ and $\dot{\theta}(t)$.

(f) Discretize the system numerically by use of the routine `c2d` in *Matlab* .

(g) Simulate the discretized system with the routine `dlsim`. Compare with the numerical solutions of the differential equation.

Exercise 2.4 (Eigenvalues and stability in the cont. and discrete case)
Origo is a stable equilibrium point for the time-invariant system $\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t)$ if and only if the eigenvalues of $\mathbf{A}$ lies in the left half of the complex plane *exclusive* the imaginary axis (from here on called the left half-plane).

(a) Consider the pendulum of Exercise 2.3. We now introduce air resistance which is assumed to influence the pendulum with a power proportional to its angle-velocity. Show that we then get the following differential equation for the movement of the pendulum:

$$L\ddot{\theta}(t) + c\dot{\theta}(t) + g\theta(t) = u(t),$$

where $c > 0$ is the friction coefficient specified by the size and surface of the pendulum ball.

(*b*) Put the system in state space form and show that the eigenvalues to the system matrix now is in the left half-plane.

(*c*) What happens with the eigenvalues when the value of c increase from 0?

(*d*) The general linear system $\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t)$ has the solution $\mathbf{x}(t) = e^{\mathbf{A}t}\mathbf{x}(0)$. Can you from this and from what you know about the eigenvalues of $e^{\mathbf{A}t}$ explain why all terms in the solution to a stable model will contain exponentially damped functions and therefore converge to zero?

(*e*) Suppose $L = 1\,\mathrm{m}$, $g = 10\,\mathrm{m/s^2}$. Use *Matlab* for simulating the system for $c = 0.5\,\mathrm{m/s}$ and some smaller and larger values. Does it fit with your physical intuition?

(*f*) Under the discretization with sampling time T we have from Exercise 2.1 the connection $\lambda_i(e^{\mathbf{A}T}) = e^{\lambda_i(\mathbf{A})T}$, $i = 1, \ldots, n$ (when $\mathbf{A}$ is $n \times n$).

Explore how the eigenvalues of $\mathbf{A}$ which are on the real or the imaginary axes are transformed under this transformation. (Put $T = 1$ for simplicity. *Matlab* calculates with complex numbers and makes plots in the complex plane.)

Exercise 2.5 (Reachability and observability)

(*a*) Consider the pendulum model in Exercise 2.4. Put again $L = 1\,\mathrm{m}$, $g = 10\,\mathrm{m/s^2}$ and $c = 0.5\,\mathrm{m/s}$. Use *Matlab* and discretize the model with $T = 0.5\,\mathrm{s}$ and show that the discrete system is reachable.

(*b*) It is possible to bring the system from any initial state to an arbitrary end state in two time step.

Find $u(0)$ and $u(1)$ such that the system is brought from $\mathbf{x}(0) = \begin{pmatrix} 0.1 & 0 \end{pmatrix}^T$ to $\mathbf{x}(2) = \begin{pmatrix} -0.1 & -0.05 \end{pmatrix}^T$.

(*c*) Assume $\mathbf{x}(0) = \begin{pmatrix} 0 & 0 \end{pmatrix}^T$. Which states can be reached in *one* time step?

(*d*) Show that the discrete system is observable both if only the angle is measured and if only the angle velocity is measured. Why is this reasonable?

(Remark: Remember that for Example 1 (one-dimensional movement without friction), the system was *not* observable if only the velocity was measured.)

(*e*) Suppose the angle $z(k) = x_1(k) = \theta(k)$ is measured and that we have observed the angles $z(0) = 0.15$, $z(1) = 0.0098$, $z(2) = -0.1169$ and $z(3) = -0.0065$. Calculate from this the initial velocity.

Exercise 2.6 (Spring system)

Consider the motion of a mass m suspended at the end of a spring and subject to an external force acting in the vertical direction. Hooke's Law will be helpful[2]. *Hooke's*

Law: The force exerted by a spring has magnitude proportional to the displacement of the spring from its equilibrium position.

A spring motion can be described as

$$m\ddot{x}(t) + ax(t) = u(t)$$

where $a > 0$ is spring constant and $u(t)$ is external force.

(a) Put the system in state space form.

(b) Do you think the system is stable? Find the eigenvalues.

(c) Find the corresponding discrete system with $T = 1$, i.e. compute $\mathbf{\Phi}$.

Since no physical spring can sustain simple harmonic motion, a frictional force must be introduced. A frictional force with frictional constant c is proportional to the velocity.

(d) Find the differential equation in this case and write the system in state space form.

(e) Analyze stability of the system for different values of c.

(f) Assume $m = 10$. What requirements for a and c are needed for the system to be reachable?

Chapter 3

Random (Gaussian) variables and processes

The purpose of this chapter is to present an elementary, informal and brief introduction to the theory of Gaussian random variables and processes with the specific need of Kalman filtering in mind. This means that the students are only assumed to have a background from an elementary course in probability theory corresponding to the work of half a semester. There is usually little emphasis on joint Gaussian distributions in such courses. The chapter will go from scratch, starting with the concept of univariate random variables. No proofs will be given. The reader is here referred to introductory textbooks in statistics and probability. Some possible references are [13] (stochastic processes) and [11] (random vectors, Gaussian distribution).

3.1 Random variables

A **random variable** x is a quantity assigned values through a probabilistic mechanism defined by a *probability density* (or the equivalent terminology *distribution function*) $p(x)$. Any non-negative function integrating to one over the real line may be a density in this sense[1]. To a density is associated its **mean** or **expectation**

$$\mu = E[x] = \int_{-\infty}^{\infty} xp(x)\,dx, \tag{3.1}$$

and its **variance**

$$\mathrm{var}[x] = E[(x-\mu)^2)] = \int_{-\infty}^{\infty} (x-\mu)^2 p(x)\,dx. \tag{3.2}$$

[1] This is not strictly true. The functions must also be what mathematicians call *measurable*. We need not bother with this technicality.

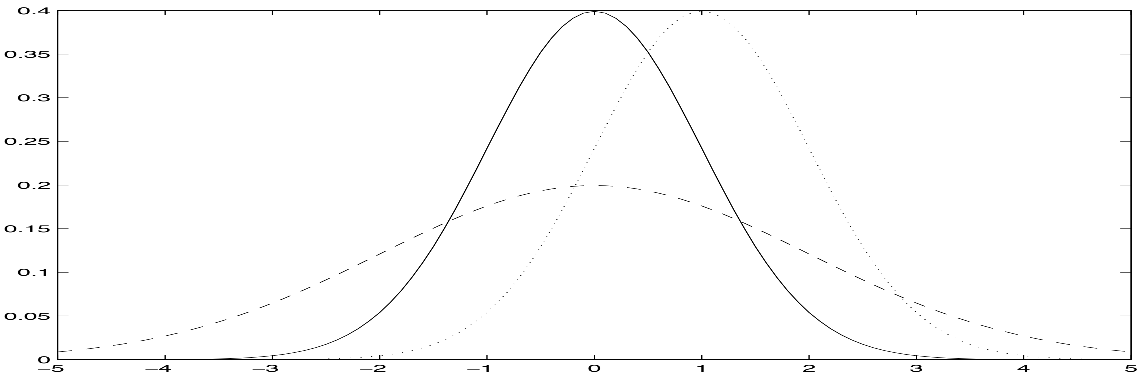

Figure 3.1: The univariate Gaussian distribution for different values of μ and σ. The solid line is for $\mu = 0$ and $\sigma = 1$, the dashed line is for $\mu = 0$ and $\sigma = 2$ while the dotted line is for $\mu = 1$ and $\sigma = 1$.

Their meaning as average and spread of the distribution is obvious from their definitions. They need not necessarily exist, i.e they could be infinite. This will in practice be no problem for the situations we will consider. The unit of the variance is the square of the unit in which x is expressed. As a result its numerical value does not admit direct interpretation. In spite of this, the variance is an immensely useful concept. The variance may be transformed to a measure on the original unit by taking the square root:

$$\text{stan}[x] = \sqrt{\text{var}[x]}. \tag{3.3}$$

This measure is denoted the **standard deviation** of the random variable x.

The most famous example, and the only one we shall consider in this chapter, is the **Gaussian** (or **normal**) density

$$p(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left\{-\frac{1}{2\sigma^2}(x-\mu)^2\right\}. \tag{3.4}$$

Here μ and $\sigma > 0$ are the *parameters* of the distribution. Their interpretation is immediate in that $\mu = E[x]$ and[2] $\sigma = \text{stan}(x)$. To specify that x has a Gaussian distribution with expectation μ and variance σ^2, we will use the notation $x \sim \mathcal{N}(\mu, \sigma^2)$.

Figure 3.1 shows $p(x)$ for different values of μ and σ. Note the symmetry around μ, while the width of the curve is determined by σ.

Expectation, variance and standard deviation are simple concepts that facilitate the interpretation of random variables. They are what the mathematicians call *functionals*, i.e. they associate a real number to a given density function. There are a number

[2] Note that μ has been used *both* as notation for expectation in (3.1) *and* as a parameter of the normal density in (3.4). The symmetry of the Gaussian density implies that the two usages are identical.

	Condition
$E[\phi(x)] = \int_{-\infty}^{\infty} \phi(x)p(x)\,dx$	None
$E[ax+b] = aE[x]+b$	None
$E[x_1 + ... + x_n] = E[x_1] + ... + E[x_n]$	None
$\mathrm{var}[ax+b] = a^2\mathrm{var}[x]$	None
$\mathrm{var}[x] = 0 \Rightarrow x = \text{constant}$	None

Table 3.1: Rules for operating means and variances. $x_1, ..., x_n$ are random variables, a and b real numbers (i.e. constants) and ϕ a real function.

of simple rules to operate them, which are convenient in probabilistic calculations. Those most needed in the following are listed in Table 3.1.

3.2 Random vectors: Pairs

The extension to *pairs* of random variables is straightforward (at least at first glance). A random vector (x_1, x_2) is defined through its *joint density function* $p(x_1, x_2)$. The only difference from the univariate case is that the density is now defined on the plane instead of on the real line. As before, any non-negative function integrating to one over both x_1 and x_2 may be used as a mathematical model for a random pair[3]. Univariate density functions for both variables may be deduced from their simultaneous one $p(x_1, x_2)$. If $p(x_1)$ and $p(x_2)$ are the densities of x_1 and x_2 *marginally*, i.e. without regard to the other[4], then

$$p(x_2) = \int_{-\infty}^{\infty} p(x_1, x_2)\,dx_1, \tag{3.5}$$

and similar for $p(x_1)$. The point in introducing joint densities is, however, a different one. When dealing with dynamic systems, there is a need for a concept that can capture stochastic *dependence* between different variables. Dependence is in the present context the same as *covariation* . To detail what this means, consider two observations that are close in time. If one is accidentally low (or high), then so is often the other. They covariate *positively.* Negative covariation is the opposite, one variable tend to be high when the other is low. The covariation is not in any fixed, deterministic sense. The dependence is modified by randomness. A powerful way of expressing the idea, is through the concept of *conditional* density. The formal definition is

$$p(x_1|x_2) = \frac{p(x_1, x_2)}{p(x_2)}, \tag{3.6}$$

[3] Subject to measurability, see preceding footnote.

[4] Densities are throughout generically denoted p, the indexing of *the argument* defining for which variables. For example, $p(x_1)$ is the density of x_1 and $p(x_1, x_2)$ the joint density of (x_1, x_2).

where we have introduced the vertical bar to indicate which variables we condition on (those on the right side of the vertical bar). Note that the conditional densities become *different functions* as x_2 is varied. It is easily verified that the integral with respect to x_1 (over the real line) is one (do it!). It is therefore a valid density, and it enables us to express how the random *variation* in x_1 changes with the *value* of x_2. It is perfectly legitimate to compute the mean and variance of this conditional distribution, for example

$$E[x_1|x_2] = \int_{-\infty}^{\infty} x_1 p(x_1|x_2)\, dx_1, \tag{3.7}$$

which is known as the **regressor** of x_1 on x_2. It will play a central role in the derivation of the Kalman filter, because it turns out that it defines how an unobserved x_1 ideally should be estimated from knowledge of x_2.

Dependence, as defined above, is a rich and complicated concept, and many applications need a more modest version. The **covariance** achieves just that. Let $\mu_1 = E[x_1]$, $\mu_2 = E[x_2]$, $P_{11} = \text{var}[x_1]$ and $P_{22} = \text{var}[x_2]$ be the mean and variances of the two random variables. Then covariance and **correlation** , which reduces dependence down to single numbers, are defined through

$$P_{12} = \text{cov}[x_1, x_2] = E[(x_1 - \mu_1)(x_2 - \mu_2)] \tag{3.8}$$

and

$$\text{corr}[x_1, x_2] = \frac{P_{12}}{(P_{11}P_{22})^{1/2}} = \frac{\text{cov}[x_1, x_2]}{\sqrt{\text{var}[x_1]\text{var}[x_2]}}. \tag{3.9}$$

Covariance is not really a dependence measure, since it is influenced by the measurement scales used (multiply everything with, say 10 and the covariance goes up by the same number). However, the denominator in (3.9) makes correlation scale-free and thus a (necessarily imperfect) way of expressing dependence as a single number.

Another important concept is that of **independence** between two random variables. That occurs when

$$p(x_1|x_2) = p(x_1), \qquad \text{for all } x_1 \text{ and } x_2. \tag{3.10}$$

This says that the density (or *variation*) of x_1 is the same whatever value of x_2. No covariation between the two is then present. Eq. (3.10) implies that

$$p(x_1, x_2) = p(x_1)p(x_2),$$

and

$$p(x_2|x_1) = p(x_2),$$

	Condition
$E[E[x_1\|x_2]] = E[x_1]$	None
$\|E[x_1 x_2]\| \leq \sqrt{E[x_1^2]E[x_2^2]}$	None
$E[\phi(x)\|x = a] = \phi(a)$	None
$E[x_1\|x_2] = E[x_1]$	Independence
$E[x_1 x_2] = E[x_1]E[x_2]$	Independence
$\mathrm{var}[x_1] = E[\mathrm{var}[x_1\|x_2]] + \mathrm{var}[E[x_1\|x_2]]$	None
$\mathrm{var}[\sum_j x_j] = \sum_j \mathrm{var}[x_j] + \sum_{j\neq k} \mathrm{cov}[x_j, x_k]$	None
$\mathrm{var}[x_1 + ... + x_n] = \mathrm{var}[x_1] + ... + \mathrm{var}[x_n]$	$x_1, ..., x_n$ uncorrelated
$\mathrm{cov}[x_1, x_1] = \mathrm{var}[x_1]$	None
$\mathrm{cov}[x_1, x_2] = \mathrm{cov}[x_2, x_1]$	None
$\mathrm{cov}[ax_1 + b, cx_2 + d] = ac \cdot \mathrm{cov}[x_1, x_2]$	None
$\mathrm{cov}[x_1 + x_2, x_3] = \mathrm{cov}[x_1, x_2] + \mathrm{cov}[x_1, x_3]$	None
$\mathrm{cov}[x_1, x_2] \leq \sqrt{\mathrm{var}[x_1]\mathrm{var}[x_2]}$	None
$\|\mathrm{corr}[x_1, x_2]\| \leq 1$	None
$\mathrm{corr}[ax_1 + b, cx_2 + d] = \mathrm{corr}[x_1, x_2]$	None
$\mathrm{corr}[x_1, x_2] = 0$	Independence

Table 3.2: Rules for dependent random variables . The independence conditions (when present) are *sufficient*, not necessary.

see (3.6). The concept of independence is thus symmetric in x_1 and x_2, as it should. Reversing the roles, i.e. starting out from x_2 given x_1 instead of x_1 given x_2 would lead to the same definition.

Mathematical properties and rules of the concepts introduced in this section, are summarized in Table 3.2. Most of these properties are easily calculated, given the definition of expectation, variance and covariance. The second property is usually called Schwarz's inequality. Note in particular the last property, which says

Proposition 3.1
Two random variables which are independent are also uncorrelated.

3.3 Covariance matrix and Gaussian random pairs

We shall in the next section deal with many random variables simultaneously and consider covariances between all pairs. It is convenient to organize these quantities in matrix form. Let us first see how this looks when there are just two variables x_1 and x_2. Define

$$\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \tag{3.11}$$

with mean vector

$$\mu = \begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix} \tag{3.12}$$

and covariances

$$\mathbf{P} = \begin{pmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{pmatrix}. \tag{3.13}$$

P, known as a **covariance matrix** of **x**, is a convenient algebraic tool. It is always symmetric, since $P_{12} = P_{21}$. **Gaussian** distributions for pairs are neatly defined from it. The density for **x** is then defined as

$$p(\mathbf{x}) = (2\pi)^{-1}|\mathbf{P}|^{-1/2}\exp\{-\frac{1}{2}(\mathbf{x}-\mu)^T\mathbf{P}^{-1}(\mathbf{x}-\mu)\}. \tag{3.14}$$

Here $\mathbf{x} = (x_1, x_2)^T$ is a column vector, and

$$|\mathbf{P}| = \begin{vmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{vmatrix} = P_{11}P_{22} - P_{12}P_{21}$$

is the determinant of **P**. The exponent in (3.14) is a quadratic form in $\mathbf{x} - \mu$, i.e. an expression containing squares and products of $x_1 - \mu_1$ and $x_2 - \mu_2$. The reader can write it out, and convince him(her)self that the expression is rather complicated. Clearly this must be even more so when additional variables are added. By contrast, the matrix form is dense and, with experience, even nice and transparent.

The Gaussian model contains five parameters. There are two in the mean vector μ, two variances P_{11} and P_{22} and the additional covariance $P_{12} = P_{21}$. It may not be obvious that the parameters in (3.14) actually have this interpretation (and that (3.14) is a density, for that matter). However, this turns out to be so. Figure 3.2 shows $p(\mathbf{x})$ for

$$\mu = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \qquad \mathbf{P} = \begin{pmatrix} 1 & \rho \\ \rho & 1 \end{pmatrix},$$

with $\rho = 0, 0.5, 0.9$. The function is symmetric around zero for $\rho = 0$, while for increasing ρ, the density is concentrated more and more around the line $x_1 = x_2$. This means that the density gives high probability for x_1 being almost equal to x_2.

Suppose $\mathbf{x} = \;\; x_1 \;\; x_2 \;)^T$ is Gaussian in a *joint* sense, i.e. with density of the form (3.14). It can then be proved that each of x_1 and x_2 are Gaussian *marginally*. Also note the structure of the density when the covariance $P_{12} = 0$. It is then seen from (3.14) that the joint density becomes the product of two univariate densities. x_1 and x_2 are therefore *independent* in this case. But from Table 3.2 (last line) independence always implies that the correlation (and hence the covariance) is 0. Thus two Gaussian variables which are uncorrelated are also independent.

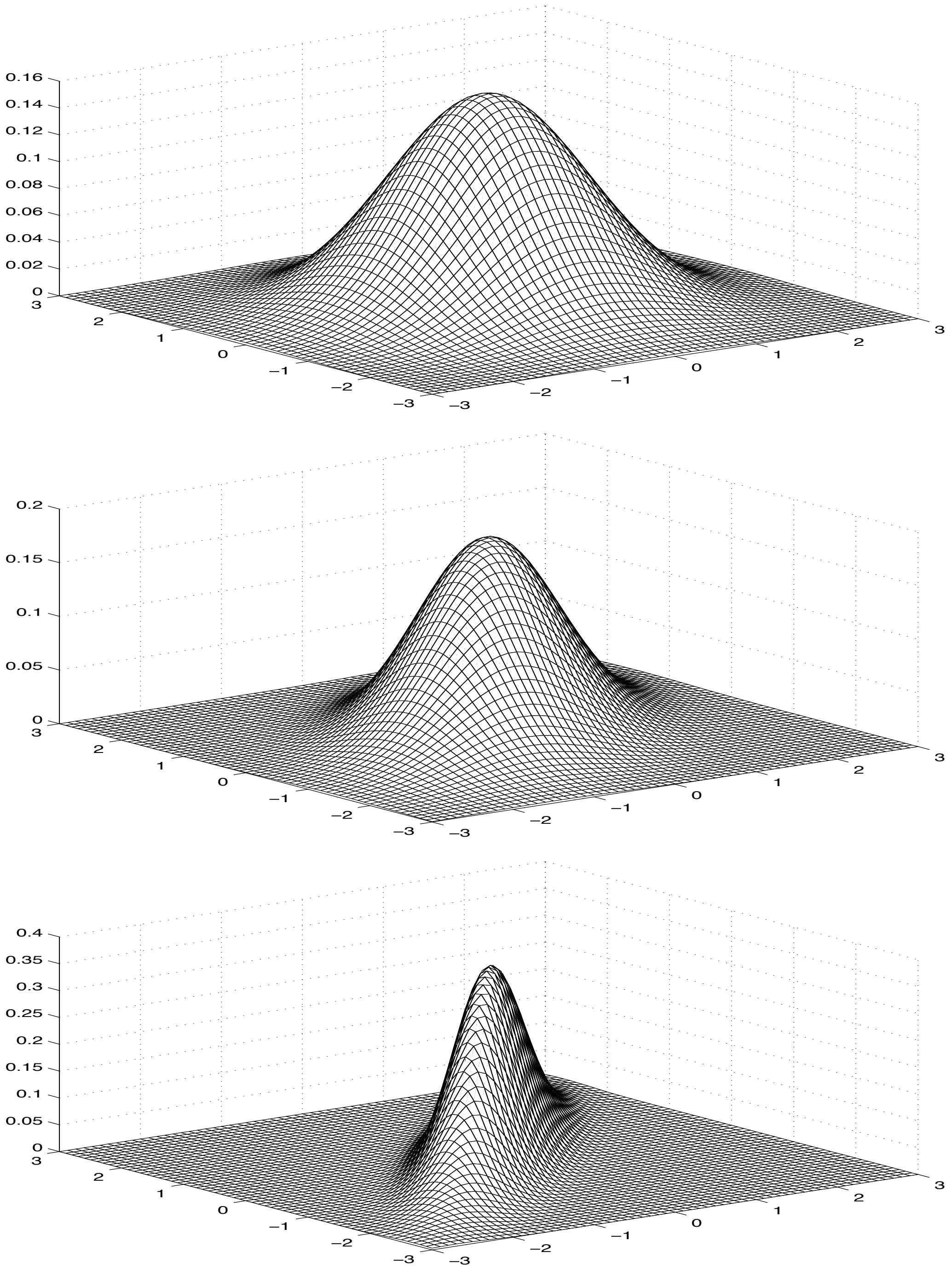

Figure 3.2: The bivariate Gaussian distribution for different correlation values ρ. The upper panel corresponds to $\rho = 0$, the middle panel to $\rho = 0.5$ and the lower panel to $\rho = 0.9$. In all cases both x_1 and x_2 have mean 0 and variance 1.

Another useful property of Gaussian models is that the conditional distribution of one variable given the other remains Gaussian. The parameters of this conditional distribution is important. Take, for example, x_1 given x_2. Then

$$E[x_1|x_2] = \mu_1 + \frac{P_{12}}{P_{22}}(x_2 - \mu_2), \tag{3.15a}$$

and

$$\text{var}[x_1|x_2] = P_{11} - \frac{P_{12}^2}{P_{22}}. \tag{3.15b}$$

These results will be proven in a more general setting in Theorem 3.2. Both formulae have intriguing interpretations. The first says that the expected value of x_1, taking the value of x_2 into account, is modified from what it would have been *without* this knowledge, by a factor proportional to $x_2 - \mu_2$. There is *no* adjustment if x_2 turned out to be what we expected ($= \mu_2$). Neither does the second factor contribute if the covariance $P_{12} = 0$. The two variables are then independent, and the second one does not contain relevant information about the first. The adjustment also becomes 0 if $P_{22} \to \infty$. Now the information in x_2 is so unreliable that it is of no value. As to (3.15b), observe that the right hand is always less than P_{11}, which is the variance of x_1 *without* knowledge of x_2. Variation (or *uncertainty*) always goes down when extra information is added. That is in the ideal world. It can be different in practice when imperfect knowledge of underlying mechanisms prevent us from using information properly (cf. Chapter 8). The formulae will be proven in a more general setting in Theorem 3.2.

Example 2
A small illustrating example with significance for dynamic systems is the following. Suppose x_1 is Gaussian with mean 10 and standard deviation 1. Let ε be another Gaussian variable, independent of the first and with mean 0 and standard deviation $\sqrt{1-a^2}$ ($|a| < 1$). Define

$$x_2 = 10 + a(x_1 - 10) + \varepsilon. \tag{3.16}$$

and assume that x_2 follows x_1 in time. From one of the results in the next section, it can be shown that x_2 is also Gaussian. From the rules in Table 3.1, we get $E[x_2] = 10$ and $\text{var}[x_2] = a^2\text{var}[x_1] + \text{var}[\varepsilon] = a^2 + (1 - a^2) = 1$, showing that x_2 *has exactly the same distribution as* x_1 *had.* Clearly we can continue this way and define a sequence of random variables with the same property. The sequence is then *stationary* in distribution (see Section 3.6 for more on this concept). The variables x_1 and x_2 are dependent. By simple calculations

$$\text{corr}[x_1, x_2] = a,$$

i.e. the degree of dependence varies with a. Eq. (3.16) is one of many stochastic models for describing covariation in time sequences. □

3.4 Random vectors: Higher dimension.

It is easy formally to extend the concept of **joint densities** (or *multivariate distributions*) to the case of n variables $\mathbf{x} = (x_1, ..., x_n)^T$. The density is a non-negative function $p(\mathbf{x})$ in $\mathbf{x}$ with integral equal to one when all n variables are integrated over the real line. Much interest in the following has to do with subvectors of $\mathbf{x}$. Let

$$\mathbf{x} = \begin{pmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{pmatrix} \begin{matrix} \}r \\ \}n-r \end{matrix}, \tag{3.17}$$

where the first subvector $\mathbf{x}_1 = (\ x_1 \ \ ... \ \ x_r\ )^T$ consists of the first r variables in $\mathbf{x}$. The second subvector is $\mathbf{x}_2 = (\ x_{r+1} \ \ ... \ \ x_n)^T$. Extensions of (3.5) and (3.6) are now

$$p(\mathbf{x}_2) = \int_{\mathbf{x}_1 \in \mathcal{R}^r} p(\mathbf{x}_1, \mathbf{x}_2) d\mathbf{x}_1, \tag{3.18}$$

which defines the **marginal** density of $\mathbf{x}_2$ by integrating over the first r variables (those in $\mathbf{x}_1$) and

$$p(\mathbf{x}_1|\mathbf{x}_2) = \frac{p(\mathbf{x}_1, \mathbf{x}_2)}{p(\mathbf{x}_2)}, \tag{3.19}$$

which is the conditional density of $\mathbf{x}_1$ given the value of $\mathbf{x}_2$. We can define **independence** between vectors as well. As in Section 3.2 this should mean that the conditional density does not depend on the set of conditioning variables, or equivalently $p(\mathbf{x}_1, \mathbf{x}_2) = p(\mathbf{x}_1)p(\mathbf{x}_2)$.

The n variables have means $\mu_j = E[x_j]$, variances $P_{jj} = \mathrm{var}[x_j]$ and covariances $P_{jk} = \mathrm{cov}[x_j, x_k]$. It is convenient to organize them in vector and matrix form, giving the **mean vector** and the **covariance matrix**, i.e. as

$$\mu = E[\mathbf{x}] = \begin{pmatrix} \mu_1 \\ \vdots \\ \mu_n \end{pmatrix}, \qquad \mathbf{P} = E[(\mathbf{x}-\mu)(\mathbf{x}-\mu)^T] \begin{pmatrix} P_{11} & \cdots & P_{1n} \\ \vdots & \ddots & \vdots \\ P_{n1} & \cdots & P_{nn} \end{pmatrix}, \tag{3.20}$$

and also to consider blocks. Introduce, consistent with the subdivision in (3.17),

$$\mu = \begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix} \begin{matrix} \}r \\ \}n-r, \end{matrix} \qquad \mathbf{P} = \begin{pmatrix} \mathbf{P}_{11} & \mathbf{P}_{12} \\ \mathbf{P}_{21} & \mathbf{P}_{22} \end{pmatrix} \begin{matrix} \}r \\ \}n-r \end{matrix} \tag{3.21}$$

Here μ_1 and μ_2 are the mean vectors of $\mathbf{x}_1$ and $\mathbf{x}_2$ and $\mathbf{P}_{11}$ and $\mathbf{P}_{22}$ their covariance matrices. The two remaining sub-matrices in (3.21) are $\mathbf{P}_{12}$ and $\mathbf{P}_{21}$. They represent covariances between the elements in $\mathbf{x}_1$ and $\mathbf{x}_2$. As mentioned in Section 3.3, all covariance matrices are *symmetric* and this makes $\mathbf{P}_{12}$ and $\mathbf{P}_{21}$ the transpose of each other, i.e. $\mathbf{P}_{12} = \mathbf{P}_{21}^T$. Independence between $\mathbf{x}_1$ and $\mathbf{x}_2$ implies that $\mathbf{P}_{12} = \mathbf{P}_{21}^T = 0$.

	Condition
$E[\mathbf{A}\mathbf{x}+\mathbf{b}] = \mathbf{A}E[\mathbf{x}]+\mathbf{b}$	None
$E[\mathbf{x}_1+...+\mathbf{x}_n] = E[\mathbf{x}_1]+...+E[\mathbf{x}_n]$	None
$E[\phi(\mathbf{x})] = \int_{-\infty}^{\infty}\phi(\mathbf{x})f(\mathbf{x})\,d\mathbf{x}$	None
$E[\phi(\mathbf{x})\vert\mathbf{x}=\mathbf{a}] = \phi(\mathbf{a})$	None
$E[E[\mathbf{x}_1\vert\mathbf{x}_2]] = E[\mathbf{x}_1]$	None
$E[\mathbf{x}_1\vert\mathbf{x}_2] = E[\mathbf{x}_1]$	Independence
$E[\mathbf{x}_1\mathbf{x}_2] = E[\mathbf{x}_1]E[\mathbf{x}_2]$	Independence
$\mathrm{VAR}[\mathbf{B}\mathbf{x}+\mathbf{a}] = \mathbf{B}\{\mathrm{VAR}[\mathbf{x}]\}\mathbf{B}^T$	None
$\mathrm{VAR}[\mathbf{x}_1+...+\mathbf{x}_n] = \mathrm{VAR}[\mathbf{x}_1]+...+\mathrm{VAR}[\mathbf{x}_n]$	$\mathbf{x}_1,...,\mathbf{x}_n$ uncorrelated
$\mathrm{VAR}[\mathbf{x}_1] = E[\mathrm{VAR}[\mathbf{x}_1\vert\mathbf{x}_2]]+\mathrm{VAR}[E[\mathbf{x}_1\vert\mathbf{x}_2]]$	None
$\mathrm{VAR}[\sum_j \mathbf{x}_j] = \sum_j \mathrm{VAR}[\mathbf{x}_j]+\sum_{j\neq k}\mathrm{COV}[\mathbf{x}_j,\mathbf{x}_k]$	None
$\mathrm{COV}[\mathbf{x}_1,\mathbf{x}_1] = \mathrm{VAR}[\mathbf{x}_1]$	None
$\mathrm{COV}[\mathbf{x}_1,\mathbf{x}_2] = \mathrm{COV}[\mathbf{x}_2,\mathbf{x}_1]$	None
$\mathrm{COV}[\mathbf{A}\mathbf{x}_1+\mathbf{b},\mathbf{C}\mathbf{x}_2+\mathbf{d}] = \mathbf{A}\{\mathrm{COV}[\mathbf{x}_1,\mathbf{x}_2]\}\mathbf{C}^T$	None
$\mathrm{COV}[\mathbf{x}_1+\mathbf{x}_2,\mathbf{x}_3] = \mathrm{COV}[\mathbf{x}_1,\mathbf{x}_2]+\mathrm{COV}[\mathbf{x}_1,\mathbf{x}_3]$	None

Table 3.3: Rules for operating mean vectors and covariance matrices. $\mathbf{x}_1,...,\mathbf{x}_n$ are random vectors, $\mathbf{a}$ a fixed vector, $\mathbf{B}$ a fixed matrix and ϕ a real function of n variables.

The opposite implication is not true, although it does hold for Gaussian distributions, as we shall see in the next section. Random vectors for which the *cross covariance* matrix $\mathbf{P}_{12}$ is 0, are called **uncorrelated**.

It will be convenient to write $E[\mathbf{x}] = \mu$ and $\mathrm{VAR}[\mathbf{x}] = \mathbf{P}$, the capital letters in 'VAR' marking the covariance matrix of a vector. Further, we will denote $\mathrm{COV}[\mathbf{x}_1,\mathbf{x}_2]$ the matrix containing as its (i,j)'th element the covariances between the i'th component of $\mathbf{x}_1$ and the j'th element of $\mathbf{x}_2$. With this convention, analogies to the operational rules in Tables 3.1 and 3.2 are set down in Table 3.3.

Note the formula requiring uncorrelatedness. A sufficient condition for this is full independence, as noted earlier.

The result about double expectation is of tremendous importance when the Kalman filter is to be derived. We will therefore prove this result explicitly.

Theorem 3.1
For any two random vectors $\mathbf{x}_1,\mathbf{x}_2$, we have (under some "general conditions") that

$$E[E[\mathbf{x}_1|\mathbf{x}_2]] = E[\mathbf{x}_1].$$

Proof of Theorem 3.1

$$
\begin{aligned}
E[\mathbf{x}_1] &= \int_{\mathbf{x}_1} \mathbf{x}_1 p(\mathbf{x}_1) d\mathbf{x}_1 \\
&\overset{(3.18)}{=} \int_{\mathbf{x}_1} \mathbf{x}_1 \int_{\mathbf{x}_2} p(\mathbf{x}_1, \mathbf{x}_2) d\mathbf{x}_2 d\mathbf{x}_1 \\
&\overset{(3.19)}{=} \int_{\mathbf{x}_1} \int_{\mathbf{x}_2} \mathbf{x}_1 p(\mathbf{x}_1|\mathbf{x}_2) p(\mathbf{x}_2) d\mathbf{x}_2 d\mathbf{x}_1 \\
&\overset{(*)}{=} \int_{\mathbf{x}_2} \int_{\mathbf{x}_1} \mathbf{x}_1 p(\mathbf{x}_1|\mathbf{x}_2) d\mathbf{x}_1 p(\mathbf{x}_2) d\mathbf{x}_2 \\
&= \int_{\mathbf{x}_2} E[\mathbf{x}_1|\mathbf{x}_2] p(\mathbf{x}_2) d\mathbf{x}_2 \\
&= E[E[\mathbf{x}_1|\mathbf{x}_2]],
\end{aligned}
$$

where the "general conditions" ensure the transition marked with a $(*)$. ■

3.5 Gaussian vectors

The only n-variate density we shall consider is the Gaussian one. The mathematical form of the density is almost as defined in (3.14). The only difference is that the constant is $(2\pi)^{-n/2}$ instead of $(2\pi)^{-1}$, and that now μ is an n-dimensional vector and $\mathbf{P}$ an $n \times n$ matrix. Similar to the univariate case, we will use the notation $\mathbf{x} \sim \mathcal{N}(\mu, \mathbf{P})$ to denote that $\mathbf{x}$ follows a Gaussian distribution with mean μ and covariance matrix $\mathbf{P}$.

Two main properties of the Gaussian model are listed in the theorem below:

Theorem 3.2
Suppose $\mathbf{x}$ *is Gaussian. Then*

(a) $\mathbf{Y} = \mathbf{A}\mathbf{x} + \mathbf{b}$ *is also Gaussian. This applies to any matrix* $\mathbf{A}$ *and any vector* $\mathbf{b}$ *for which the operations are defined.*

(b) *Divide* $\mathbf{x}$ *into subvectors* $\mathbf{x}_1$ *and* $\mathbf{x}_2$*. Then the conditional distribution of* $\mathbf{x}_1$ *given* $\mathbf{x}_2$ *(and* $\mathbf{x}_2$ *given* $\mathbf{x}_1$*) is also Gaussian with mean and covariance matrix given by*

$$E[\mathbf{x}_1|\mathbf{x}_2] = \mu_{1|2} = \mu_1 + \mathbf{P}_{12}\mathbf{P}_{22}^{-1}(\mathbf{x}_2 - \mu_2), \tag{3.22a}$$

and

$$VAR[\mathbf{x}_1|\mathbf{x}_2] = \mathbf{P}_{1|2} = \mathbf{P}_{11} - \mathbf{P}_{12}\mathbf{P}_{22}^{-1}\mathbf{P}_{21}. \tag{3.22b}$$

Note that the formulae (3.22) are generalizations of the formulae (3.15). The reder should verify that (3.15) are special cases.

Both results need filling out. Take the first one. The parameters of the new Gaussian vector $\mathbf{Y}$ are $E[\mathbf{Y}] = \mathbf{A}\mu + \mathbf{b}$ and $\mathrm{VAR}[\mathbf{Y}] = \mathbf{A}\{\mathrm{VAR}[\mathbf{x}]\}\mathbf{A}^T$, both directly obtained from Table 3.3. There are many interesting special cases to that result. Here are two: If $\mathbf{A} = (1, 1, 1, ..., 1)$ is a row vector of 1's and $\mathbf{b} = 0$, then $\mathbf{Y} = x_1 + ... + x_n$, i.e. the sum of Gaussian variables are always Gaussian. Secondly, take as $\mathbf{A}$

$$\mathbf{A} = \begin{pmatrix} \mathbf{I}_r & \mathbf{0} \end{pmatrix}\begin{matrix} \}r \\ \}n-r, \end{matrix}$$

where $\mathbf{I}_r$ is the identity matrix of size $r \times r$ (i.e. the matrix with 1 on the main diagonal and 0 elsewhere) and the 0's are matrices consisting only of $\mathbf{0}$'s. If $\mathbf{b} = 0$ as before, then $\mathbf{Y} = \mathbf{x}_1$ where $\mathbf{x}_1$ is the subvector of $\mathbf{x}$ defined earlier. But we have now proved that subvectors of Gaussian vectors must themselves be Gaussian.

In order to prove Theorem 3.2, we will need the following lemma:

Lemma 3.1
Assume $(\mathbf{x}_1, \mathbf{x}_2)$ are jointly Gaussian distribution. If $\mathbf{x}_1$ and $\mathbf{x}_2$ are uncorrelated, they are also independent.

Proof of Lemma 3.1 The two variables $\mathbf{x}_1, \mathbf{x}_2$ are independent if $p(\mathbf{x}_1, \mathbf{x}_2) = p(\mathbf{x}_1)p(\mathbf{x}_2)$. Now, since $\mathbf{P}_{12} = \mathbf{0}$, we have $|\mathbf{P}| = |\mathbf{P}_{11}||\mathbf{P}_{22}|$. Further

$$\mathbf{P}^{-1} = \begin{pmatrix} \mathbf{P}_{11}^{-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{P}_{22}^{-1} \end{pmatrix}$$

which leads to

$$\begin{aligned} &(\mathbf{x} - \mu)\mathbf{P}^{-1}(\mathbf{x} - \mu)^T \\ &\qquad = (\mathbf{x}_1 - \mu_1)\mathbf{P}_{11}^{-1}(\mathbf{x}_1 - \mu_1)^T + (\mathbf{x}_2 - \mu_2)\mathbf{P}_{22}^{-1}(\mathbf{x}_2 - \mu_2)^T, \end{aligned}$$

resulting in that $p(\mathbf{x}_1, \mathbf{x}_2) = p(\mathbf{x}_1)p(\mathbf{x}_2)$. ■

Proof of Theorem 3.2 For the first part of the theorem, we refer to [11]. Concerning the second result, we will give a proof thereof, based on the first result. The subsequent argument is taken from [11].

Recall that each of the subvectors $\mathbf{x}_1$ and $\mathbf{x}_2$ are Gaussian, as proven above. Introduce

$$\begin{aligned} \mathbf{x}_{1|2} &= \mathbf{x}_1 - \mathbf{P}_{12}\mathbf{P}_{22}^{-1}\mathbf{x}_2 \\ &= \begin{pmatrix} \mathbf{I}_r, & -\mathbf{P}_{12}\mathbf{P}_{22}^{-1} \end{pmatrix}\begin{pmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{pmatrix}. \end{aligned} \tag{3.23}$$

Since $\mathbf{x}_{1|2}$ is a linear transformation of $\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2)^T$, it must be Gaussian. The mean vector and covariance matrix are, according to Table 3.3

$$E[\mathbf{x}_{1|2}] = \mu_1 - \mathbf{P}_{12}\mathbf{P}_{22}^{-1}\mu_2 \tag{3.24}$$

and

$$\begin{aligned}\mathrm{VAR}[\mathbf{x}_{1|2}] =&(\mathbf{I}_r, -\mathbf{P}_{12}\mathbf{P}_{22}^{-1})\mathbf{P}(\mathbf{I}_r, -\mathbf{P}_{12}\mathbf{P}_{22}^{-1})^T \\ =&(\mathbf{I}_r, -\mathbf{P}_{12}\mathbf{P}_{22}^{-1})\begin{pmatrix} \mathbf{P}_{11} & \mathbf{P}_{12} \\ \mathbf{P}_{21} & \mathbf{P}_{22} \end{pmatrix}\begin{pmatrix} \mathbf{I}_r \\ -\mathbf{P}_{22}^{-1}\mathbf{P}_{21} \end{pmatrix},\end{aligned}$$

since $(\mathbf{P}_{22}^{-1})^T = \mathbf{P}_{22}^{-1}$ and $\mathbf{P}_{12}^T = \mathbf{P}_{21}$. Thus

$$\mathrm{VAR}[\mathbf{x}_{1|2}] = \mathbf{P}_{11} - \mathbf{P}_{12}\mathbf{P}_{22}^{-1}\mathbf{P}_{21}, \tag{3.25}$$

after blockwise matrix multiplication. The cross covariance matrix $\mathrm{COV}[\mathbf{x}_{1|2}, \mathbf{x}_2]$ is interesting. Indeed,

$$\mathrm{COV}[\mathbf{x}_{1|2}, \mathbf{x}_2] = E[(\mathbf{x}_{1|2} - E[\mathbf{x}_{1|2}])(\mathbf{x}_2 - \mu_2)^T],$$

and after $\mathbf{x}_{1|2}$ and $E[\mathbf{x}_{1|2}]$ are replaced by (3.23) and (3.24)

$$\begin{aligned}\mathrm{COV}[\mathbf{x}_{1|2}, \mathbf{x}_2] =&E[((\mathbf{x}_1 - \mu_1) - \mathbf{P}_{12}\mathbf{P}_{22}^{-1}(\mathbf{x}_2 - \mu_2))(\mathbf{x}_2 - \mu_2)^T] \\ =&E[(\mathbf{x}_1 - \mu_1)(\mathbf{x}_2 - \mu_2)^T] - \\ &\quad \mathbf{P}_{12}\mathbf{P}_{22}^{-1}E[(\mathbf{x}_2 - \mu_2)(\mathbf{x}_2 - \mu_2)^T] \\ =&\mathbf{P}_{12} - \mathbf{P}_{12}\mathbf{P}_{22}^{-1}\mathbf{P}_{22} \\ =&\mathbf{0}.\end{aligned}$$

Hence:

Lemma 3.2
$\mathbf{x}_{1|2}$, *defined in* (3.23) *is uncorrelated with* $\mathbf{x}_2$, *thus in the Gaussian case also independent.*

The lemma yields a simple derivation for the conditional distribution of $\mathbf{x}_1$ given $\mathbf{x}_2$. Turn (3.23) (first line) around so that it reads

$$\mathbf{x}_1 = \mathbf{P}_{12}\mathbf{P}_{22}^{-1}\mathbf{x}_2 + \mathbf{x}_{1|2}. \tag{3.26}$$

Suppose $\mathbf{x}_2$ *is fixed.* The first term on the right in (3.26) is then simply a constant, while the other term $\mathbf{x}_{1|2}$, which is independent of $\mathbf{x}_2$ has the Gaussian distribution stated above, i.e. with mean and covariance matrix as given in (3.24) and (3.25). Thus $\mathbf{x}_1$ must, for fixed $\mathbf{x}_2$, also be Gaussian distributed. The mean vector is $\mathbf{P}_{12}\mathbf{P}_{22}^{-1}\mathbf{x}_2 + E[\mathbf{x}_{1|2}]$ and the covariance matrix is equal to the covariance matrix of $\mathbf{x}_{1|2}$. Hence, after a trivial rearrangement, inserting (3.24) for $\mu_{1|2}$, we have proven the second part of Theorem 3.2. ■

An important corollary of Theorem 3.2 is the following

Corollary 3.1
Assume $\mathbf{x} = (\mathbf{x}_1^T, \mathbf{x}_2^T, \mathbf{x}_3^T)^T$ *where* $corr[\mathbf{x}_2, \mathbf{x}_3] = \mathbf{0}$. *Then*

$$\begin{aligned} E[\mathbf{x}_1|\mathbf{x}_2, \mathbf{x}_3] =& E[\mathbf{x}_1|\mathbf{x}_2] + \mathbf{P}_{13}\mathbf{P}_{33}^{-1}(\mathbf{x}_3 - \mu_3) \\ VAR[\mathbf{x}_1|\mathbf{x}_2, \mathbf{x}_3] =& VAR[\mathbf{x}_1|\mathbf{x}_2] - \mathbf{P}_{13}\mathbf{P}_{33}^{-1}\mathbf{P}_{31} \end{aligned}$$

where $\mu_i = E[\mathbf{x}_i]$ *and* $\mathbf{P}_{ij} = COV[\mathbf{x}_i, \mathbf{x}_j]$.

Proof Note first that

$$\begin{aligned} E[\mathbf{x}_1|\mathbf{x}_2, \mathbf{x}_3] =& \mu_1 + (\mathbf{P}_{12}, \mathbf{P}_{13}) \begin{pmatrix} \mathbf{P}_{22} & \mathbf{0} \\ \mathbf{0} & \mathbf{P}_{33} \end{pmatrix}^{-1} \begin{pmatrix} \mathbf{x}_2 - \mu_2 \\ \mathbf{x}_3 - \mu_3 \end{pmatrix} \\ =& \mu_1 + \mathbf{P}_{12}\mathbf{P}_{22}^{-1}(\mathbf{x}_2 - \mu_2) + \mathbf{P}_{13}\mathbf{P}_{33}^{-1}(\mathbf{x}_3 - \mu_3) \end{aligned}$$

However, from Theorem 3.2, the first two terms on the right hand side is equal to $E[\mathbf{x}_1|\mathbf{x}_2]$, proving the first part of the Corollary.

The second part is shown similarly:

$$\begin{aligned} \text{VAR}[\mathbf{x}_1|\mathbf{x}_2, \mathbf{x}_3] =& \mathbf{P}_{11} + (\mathbf{P}_{12}, \mathbf{P}_{13}) \begin{pmatrix} \mathbf{P}_{22} & \mathbf{0} \\ \mathbf{0} & \mathbf{P}_{33} \end{pmatrix}^{-1} \begin{pmatrix} \mathbf{P}_{21} \\ \mathbf{P}_{31} \end{pmatrix} \\ =& \mathbf{P}_{11} + \mathbf{P}_{12}\mathbf{P}_{22}^{-1}\mathbf{P}_{21} + \mathbf{P}_{13}\mathbf{P}_{33}^{-1}\mathbf{P}_{31} \\ =& \text{VAR}[\mathbf{x}_1|\mathbf{x}_2] + \mathbf{P}_{13}\mathbf{P}_{33}^{-1}\mathbf{P}_{31} \end{aligned}$$

again by Theorem 3.2. ■

3.6 Random processes

A **random process** is a collection of random variables $\{x(t)\}$, indexed in time. If plotted against t, the process will describe a stochastic curve with random disturbances superimposed on local and global trends. The practical meaning entails that the individual variables $x(t)$ and $x(s)$ are stochastically dependent. In particular, if t and s are close in time, we would expect their values to be highly correlated. The purpose of this section is to introduce the most important concepts for random processes.

Suppose $x(t)$ is defined on an interval from 0 to T. It is possible to regard this as an extension of the situation in the preceding section where n random variables were considered jointly. Now, however, the situation has become even more complicated since there are an infinite number of random variables to take care of [5]. We can not specify the distribution for more than a finite number of variables. Fortunately this is enough for a consistent definition. The distribution of a random process is defined

[5] The number is not even *countable* since $[0, T]$ is an interval.

by taking points $t_1 < t_2 < ... < t_n$ and specifying a distribution for the corresponding variables $x(t_1), x(t_2), ..., x(t_n)$. This has to be done for all $n = 1, 2, ..$ and all collections of $t_1, ..., t_n$ inside $[0, T]$. It is not obvious that this can be done at all. At least there must be certain consistency requirements. For example, the density of $x(t_1)$ follows from the joint one for $x(t_1), x(t_2)$ (see Section 3.2), and the two densities can not be specified independent of each other. The problem is solved in probability theory, and necessary and sufficient conditions for consistency have been worked out. We shall take this for granted. Usually the process is defined through some random mechanism. An example is given in (3.29) below.

An important concept is that of stationarity. A process $\{x(t)\}$ is called **stationary** if *the joint distribution of* $x(t_1 + s), x(t_2 + s), ..., x(t_n + s)$ *does not depend on* s. This is to hold for all n and all $t_1, ..., t_n$. The definition says that the probability distribution does not change in time. If the *time differences* are the same in a future collection of random variables, the joint distribution remains the same. In particular, all $x(t)$ must for a stationary random process have the same distribution. The mean $\mu = E[x(t)]$ is therefore for such processes a *constant* and the same goes for the variance. Also note the implication for covariances. If $c(t_1, t_2) = \text{cov}[x(t_1), x(t_2)]$ is the covariance of $x(t_1), x(t_2)$, then $c(t_1, t_2) = c(t_1 + s, t_2 + s)$ under stationarity. Take, in particular, $s = -t_1$, Then $c(t_1, t_2) = c(0, t_2 - t_1)$ and the covariance for a stationary process depends on the *time difference* only.

In practice, a process can only be observed at discrete time points, for example at $t_k = k\delta$, $k = 0, 1, 2, ...$ The stochastic process is then $\{x(k\delta)\}$ which we shall denote $\{x(k)\}$ from now on. The longer the increment δ, the more information is lost by going from the continuous to the sampled case, but we shall not go into this.

Consider the process $\{x(k)\}$. Let $\mu(k) = E[x(k)]$ be its mean and define the covariance to be $c(k, l) = \text{cov}[x(k), x(l)]$. If the process is stationary, $\mu(k) = \mu$ is a constant and $c(k, l) = c(l - k)$ depends only on $l - k$. Stationarity is in practical applications often used in this less stringent sense[6].

The function

$$c(\tau) = \text{cov}[x(k), x(k + \tau)], \qquad \text{valid under stationarity,} \tag{3.27}$$

is called the **autocovariance function** of the process $\{x(k)\}$. The function $c(\tau)$ describes the covariance for variables that are τ time units apart. Note that $c(0) = \text{var}[x(k)]$. Hence

$$\rho(\tau) = \frac{c(\tau)}{c(0)} \tag{3.28}$$

is the corresponding *correlation* (or autocorrelation). $\{\rho(\tau)\}$ is known as the **autocorrelation function**.

[6] The former definition is often referred to as *strict* or *strong* stationarity. If it is only demanded that $\mu(k) = \mu$ and that $c(k, l) = c(k - l)$, the process is **weakly stationary**.

Although dependence (or correlation) usually is an important ingredient in random processes, *uncorrelated* processes are important as *building blocks* for other models. A process containing uncorrelated identically distributed variables is called a **white noise** process , in contrary to a process with dependent variables which is called a colored process.

A random process where $x(t_1), x(t_2), ..., x(t_n)$ is a Gaussian vector for all n and $t_1, ..., t_n$ is called a *Gaussian random process* . Such processes will be the only ones we will consider.

Example 2 (cont.)
A natural extension of the example in Section 3.3 is,

$$x(k) = ax(k-1) + w(k-1), \qquad k = 1, 2... \tag{3.29}$$

Here a is a fixed coefficient and $\{w(k)\}$ is a white noise process. The term means that $E[w(k)] = 0$ for all k, the variance is constant and that all autocovariances are 0. *Gaussian* white noise is a process of *independent* random variables. Its autocorrelation function is $\rho(0) = 1$ and $\rho(\tau) = 0$ for $\tau \neq 0$. Take in (3.29) $\text{var}[w(k)] = \sigma^2(1-a^2)$ and suppose $\{w(k)\}$ is Gaussian. Also assume, at the start, that $x(0)$ is normal with mean 0 and variance σ^2. It then can be shown that $\{x(k)\}$ is a stationary Gaussian process with mean 0 and autocovariance function

$$c(\tau) = \sigma^2 a^\tau, \qquad \tau = 0, 1, 2, ... \tag{3.30}$$

The corresponding autocorrelation function becomes

$$\rho(\tau) = a^\tau, \qquad \tau = 0, 1, 2, ... \tag{3.31}$$

This example is known as the **autoregressive** process of order one and also as the *Gauss-Markov* process. We shall derive the formulae (3.30) and (3.31) in one of the exercises. □

3.7 Problems

Exercise 3.1 (Proving some rules on expectation and variance)

(*a*) Based on $E[\phi(x)] = \int_{-\infty}^{\infty} \phi(x)p(x)\,dx$, show that $E[ax+b] = aE[x] + b$ and $\text{var}[ax+b] = a^2\text{var}[x]$.

(*b*) Prove Proposition 3.1

Exercise 3.2 (Proving rules on covariance)

(*a*) Show that $\text{cov}[ax_1 + b, cx_2 + d] = ac \cdot \text{cov}[x_1, x_2]$.

(*b*) Show that $\mathrm{corr}[ax_1 + b, cx_2 + d] = \mathrm{corr}[x_1, x_2]$.

Exercise 3.3 (Conditional Gaussian distribution)
By using the definition of the conditional distribution, show the equations for conditional mean and variance in the Gaussian distribution given by (3.15).

(Hint: Show that the conditional distribution is on the form (3.4).)

Exercise 3.4 (Autocorrelation)
Prove the formulae for autocovariance (3.30) and autocorrelation (3.31) assuming the model (3.29).

Exercise 3.5 (Random numbers?)
Simulation of random variables is important for exploring the properties of a model. An important part of the simulation approach is the ability to perform simulations from a given distribution. Dependent variables can in many cases be constructed by transformation from independent variables, making the generation of independent variables an important building block.

Generation of *exact* independent random variables is an impossible task. One therefore has to stick to *pseudo-random* variables, which are only approximately following the desired distribution and only approximately independent. In particular, when generating many variables, the variables are constructed in sequence following a deterministic rule. We will in this exercise explore the behavior of our random generator for Gaussian distributions.

(*a*) Simulate 100 variables $x_1, ..., x_{100}$ from the standard Gaussian distribution (the Gaussian distribution with expectation zero and variance equal to 1).

Make a histogram of the variables. Does it look reasonable? (You may also try higher number of variables.)

(*b*) Make a plot of x_k against x_{k+1}. Do you see any dependence between x_k and x_{k+1}? Why is this a good check on the behavior of the random generator?

(*c*) The number generators in *Matlab* which generate random numbers use a so-called **seed**. This seed can be specified by the command `randn('seed',`⟨*num*⟩`)` in *Matlab* .

Generate a vector of length 100 with numbers from `randn` so that each number is generated with its own index as seed. That is $x(k)$ is generated with `randn('seed',`*k*`)`. Plot this vector. Comment on the result.

Exercise 3.6 (Estimation of expectation and variance)
Assume $x \sim \mathcal{N}(\mu, \sigma^2)$. In most cases we are in a situation where μ (the expectation) and σ^2 (the variance) are unknown parameters. Assume however that we have observed $x(1), ..., x(N)$ which all follow the same Gaussian distribution as x above.

There is in this case possible to *estimate* the unknown parameters based on the observations. In particular, estimates for μ and σ^2 are given by the formulae

$$\hat{\mu} = \frac{1}{N}\sum_{k=1}^{N} x(k), \qquad \hat{\sigma}^2 = \frac{1}{N}\sum_{k=1}^{N}(x(k)-\hat{\mu})^2.$$

How these estimators are actually obtained will be discussed further in the following chapters.

(a) Write a routine in *Matlab* which has as input μ, σ and N, and first simulates N independent variables from $\mathcal{N}(\mu, \sigma^2)$, and thereafter calculates estimates of μ and σ^2 based on the formulae above.

(b) Run the routine you wrote in (a) 20 times and store the estimates for μ and σ^2 in two vectors of length 20. Use these vector's to evaluate the properties of the estimators (i.e are they unbiased, what are their uncertainty).

(c) Using the definitions and rules for expectations and variances, calculate the expectation and variance of $\hat{\mu}$.

(d) Try also to calculate the expectation of $\hat{\sigma}^2$.
(Hint: Rewrite $x(k) - \hat{\mu}$ to $x(k) - \mu + \mu - \hat{\mu}$ in the formula for $\hat{\sigma}^2$.)

Exercise 3.7 (Estimation of the autocorrelation function)
A time series is a sequence of stochastic (random) variables $x(k)$ for $k = 1, \ldots, N$. The variables may very well be dependent. The autocorrelation function $\rho(\tau)$ for $x(k)$ is a measure of the linear dependence between $x(k)$ and $x(k+\tau)$ for $\tau = 1, \ldots, N-k$. Autocorrelation is the same as correlation, the name is different only to explicitly mark that the correlation is between two variables in the same time series. The definition is given in (3.28). Assuming stationarity, the autocorrelation does not depend on which positions, k and $k+\tau$, of the time series it is measured, but only on the distance (τ) between the two variables. The autocorrelation is estimated by

$$\hat{\rho}(\tau) = \frac{\frac{1}{N}\sum_{k=1}^{N-\tau}(x(k)-\bar{x}_N)(x(k+\tau)-\bar{x}_N)}{\frac{1}{N}\sum_{k=1}^{N}(x(k)-\bar{x}_N)^2}, \quad \tau = 0, \ldots, N-1$$

where $\bar{x}_N = \frac{1}{N}\sum_{k=1}^{N} x(k)$.

(a) Implement a routine in *Matlab* for calculating the autocorrelation function for a given time series. Try to utilize the possibilities of vectorization.[7]

[7] Rather than using a `for`-loop for calculating a sum of products, you can f. ex. use `sum(x(a:b-1).*x(a+1:b))`. Write `help arith` in *Matlab* if you haven't done this before. The *Matlab* -code will then both be easier to read and faster to execute.

(b) Generate a white noise process $\{x_1(k)\}$ where $x_1(k) \sim \mathcal{N}(0,1)$ for $k = 0, ..., 100$. Then construct two other processes $\{x_2(k)\}$ and $\{x_3(k)\}$ by the transformations

$$\begin{aligned} x_2(1) &= x_1(0), \quad x_2(k) = \sqrt{0.5}x_2(k-1) + \sqrt{0.5}x_1(k-1), \quad && k = 2, ..., 100 \\ x_3(1) &= x_1(0), \quad x_3(k) = \sqrt{0.5}x_1(k-1) + \sqrt{0.5}x_1(k). && k = 2, ..., 100 \end{aligned}$$

$\{x_2(k)\}$ is an example of a so-called *autoregressive* (AR) process, while $\{x_3(k)\}$ is an example of a *moving average* (MA) process. Both types of process will be considered further in Chapter 7.

(c) Estimate the autocorrelation functions for each of the time series and plot these as well as the series themselves. Comment on the results.

(d) Plot $x(k+1)$ against $x(k)$ for $k = 1, \ldots, N-1$ (that is $x(k+1)$ as a function of $x(k)$). Comment on the results.

(e) For all the three processes, calculate analytically the autocorrelation functions and compare with your results from (c).

Chapter 4

The general stochastic state space model

4.1 Introduction

In Chapter 2, a discrete time deterministic state space model was introduced in (2.7). In this chapter, we will discuss the stochastic version of the state model.

Consider a collection of random variables $x(k)$, $k = 0, 1, 2, ..$ associated with time points t_k, which may be equidistant (so that $t_k = k\delta$ for some common δ) or they may not. Such a set of random variables is a random process as defined in Section 3.6, the idea being that there exist underlying dynamic forces linking $x(k)$ and $x(j)$ at different time points t_k and t_j, although not in a completely foreseeable way. There will in engineering frequently be possible to construct dynamic relationships on physical grounds, but there remains almost always some part of the problem that can *not* be modeled. That part automatically enters the error terms which are often formulated stochastically as we shall do here.

Example 3 (random walk)
We start by introducing the simplest linear stochastic state space model conceivable. Assume the very special model

$$x(k) = x(k-1) + w(k-1), \tag{4.1}$$

where $w(k-1)$, $k =, 1, 2, ...$ are *uncorrelated* random variables with zero mean and variances

$$\text{var}[w(k)] = Q(k). \tag{4.2}$$

The process $x(k)$ defined in (4.1) is called a **random walk model** and is much used in statistics. The name stands from the fact that the process at t_k is the same as it

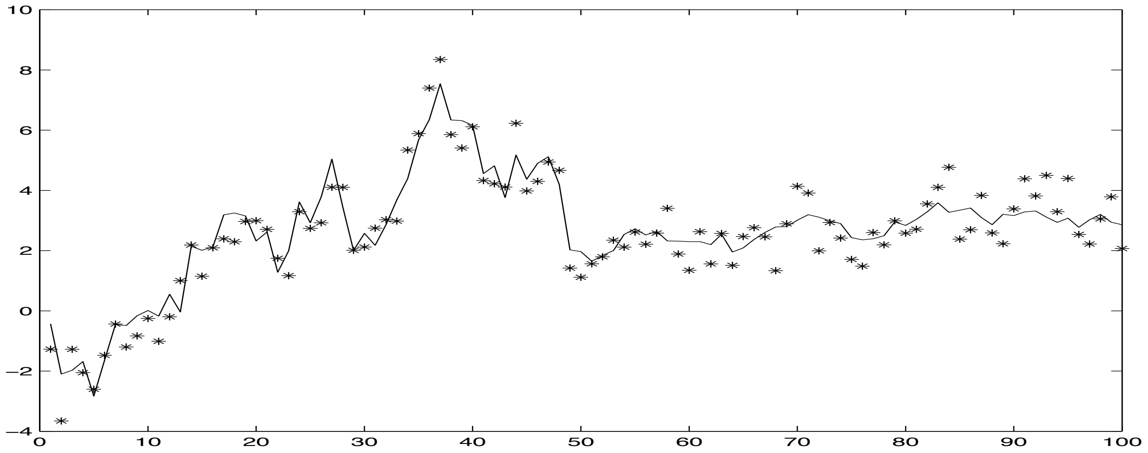

Figure 4.1: Simulation of the random walk process. $Q(k-1) = 1$ for $k = 1, ..., 50$ and $= 0.1$ for $k = 51, ..., 100$. Observations are simulated with $R(k) = 0.5$ for all k. The process is displayed by the solid line (linear interpolation). The observations are superimposed as dots.

was at t_{k-1}, except for the random term $w(k-1)$ which is as likely to add to as to subtract from the former value $x(k-1)$. There is *no drift* in the process, and it can be proved by the results in Chapter 3 that, if all $w(k)$ are stochastically independent, then the mean of $x(k)$ *if we know* $x(k-1)$ is $x(k-1)$.

The process has been simulated in figure 4.1 (solid line) for two values of the variance $Q(k)$ (*Matlab* commands are given in Appendix A). Note that the trace may fluctuate wildly or be quite smooth. Large values of $Q(k)$ gives birth to unstable processes, differing a lot from one time point to the next one, whereas small values lead to smoothness. A special case worth mentioning is when $Q(k) = 0$. Eq. (4.1) then reduces to $x(k) = x(k-1)$, and the "process" $\{x(k)\}$ is a constant. Although this case may at this point seem trivial, we shall see later that many important statistical models are actually conceived in this way.

The preceding x-process (and its extension in the next section) is one of two ingredients in a linear state space model. The other element is a relationship connecting $x(k)$ to its actual observation $z(k)$, allowing for error or noise in the measurement. Suppose, for example, that

$$z(k) = x(k) + v(k), \tag{4.3}$$

where $v(k)$ is the error term. It is assumed that $v(k)$ is a process of the same type as the w-process in (4.1), i.e uncorrelated with zero mean and variance

$$\mathrm{var}[v(k)] = R(k). \tag{4.4}$$

It is assumed in addition that the two processes $\{w(k)\}$ and $\{v(k)\}$ are mutually uncorrelated (i.e. $\mathrm{corr}[w(k), v(j)] = 0$ for all k and j) or, sometimes a little stronger, stochastically independent. The implication is that the random disturbances affecting the measurements have nothing to do with the randomness in the process $\{x(k)\}$ itself. This is surely plausible under many circumstances, since it means that the dynamics and the measurements are influenced by different, independent factors. An additional commonly assumed assumption is that $w(k)$ and $v(k)$ should be uncorrelated with $x(0)$.

Measurements (simulated) are marked with dots in Figure 4.1. Note how the variation around the solid curves are linked to the size of the variance $R(k)$. We shall see later, when trying to recover the x-process from the measurements, that what really matters is the ratio of the two variances (4.2) and (4.4). This restoration problem is a central one. □

We shall refer to $w(k-1)$ as the *error* or *noise* at time t_k, since it adds to the dynamics (here the trivial $x(k) = x(k-1)$) an element that can only be understood in probabilistic terms. The total collection $\{w(k)\}$ of noise variables is called a **noise process**. Such noise processes are essential tools in describing the randomness in the phenomena under study. The symbols $w(k)$ and $v(k)$ will be used to designate noise processes that

- have mean 0, i.e. for example $E[w(k)] = 0$,
- are uncorrelated, i.e. $\text{corr}[w(k), w(j)] = 0$ for $k \neq j$
- and have magnitude described by their variances, i.e. $\text{var}[w(k)] = Q(k)$.

A noise process with uncorrelated variables is usually called a *white noise* process.

In the example above, we assumed there was no *structure* in the error sequences $\{w(k)\}$ and $\{v(k)\}$. This is a standard assumption made for such models. In some cases there may however be some structure present meaning that errors at neighboring points in time have a tendency to be similar (or, less common, dissimilar). We shall see in Chapter 7 how such effects, often called *autocorrelations*, can be formulated mathematically and put into the class of linear state space models which utilize as building blocks *uncorrelated* processes in the sense defined above. Uncorrelatedness can be regarded as an independence concept, although weaker than full stochastic independence that demands all $w(k)$ to be independent. The two notions become equivalent in case of Gaussian processes (see Chapter 3), an assumption we shall make from time to time throughout.

4.2 The general stochastic state space model

The general **stochastic state model** is obtained from the deterministic model in eq. (2.7) by addition of process and measurement noise.

$$\begin{aligned}\mathbf{x}(k) &= \mathbf{\Phi}(k-1)\mathbf{x}(k-1)+\mathbf{\Psi}(k-1)\mathbf{u}(k-1)+\mathbf{w}(k-1), && \text{(system)}, && (4.5a)\\ \mathbf{z}(k) &= \mathbf{H}(k)\mathbf{x}(k)+\mathbf{v}(k), && \text{(observations)}. && (4.5b)\end{aligned}$$

In much of the mathematical derivations to follow, the presence of the input functions will complicate the matter without giving extra insight into the problem. We will therefore mainly be working with the more simple model neglecting deterministic inputs:

$$\begin{aligned}\mathbf{x}(k) &= \mathbf{\Phi}(k-1)\mathbf{x}(k-1)+\mathbf{w}(k-1), && \text{(system)}, && (4.6a)\\ \mathbf{z}(k) &= \mathbf{H}(k)\mathbf{x}(k)+\mathbf{v}(k), && \text{(observations)} && (4.6b)\end{aligned}$$

where $k = 1, 2, \ldots$ and $\mathbf{\Phi}(k)$ and $\mathbf{H}(k)$ are deterministic coefficient matrices. The **state vector** $\mathbf{x}(k)$ and the **observation vector** $\mathbf{z}(k)$ are vectors of size n and m, respectively, implying that $\mathbf{\Phi}(k)$ and $\mathbf{H}(k)$ are matrices of dimensions $n\times n$ and $m\times m$. The main problem we will consider is how to estimate the state vector based on the observations.

The noise processes $\{\mathbf{w}(k)\}$ and $\{\mathbf{v}(k)\}$ are assumed to have the following properties:

$$\text{COV}[\mathbf{w}(k),\mathbf{w}(k')] = \begin{cases}\mathbf{Q}(k), & k=k'\\ \mathbf{0} & k\neq k'\end{cases} \qquad (4.7a)$$

$$\text{COV}[\mathbf{v}(k),\mathbf{v}(k')] = \begin{cases}\mathbf{R}(k), & k=k'\\ \mathbf{0}, & k\neq k'\end{cases} \qquad (4.7b)$$

which means that both $\{\mathbf{w}(k)\}$ and $\{\mathbf{v}(k)\}$ are white noise process, and

$$\begin{aligned}\text{COV}[\mathbf{w}(k),\mathbf{v}(k')] &= \mathbf{0} && (4.7c)\\ \text{COV}[\mathbf{x}(0),\mathbf{v}(k)] &= \mathbf{0} && (4.7d)\\ \text{COV}[\mathbf{x}(0),\mathbf{w}(k)] &= \mathbf{0} && (4.7e)\end{aligned}$$

which state that the two noise processes are uncorrelated with each other and uncorrelated with the initial state.

The general model (4.6) reduces to a **time-invariant** if the parameters $\{\mathbf{\Phi}(k)\}$, $\{\mathbf{H}(k)\}$, $\{\mathbf{Q}(k)\}$ and $\{\mathbf{R}(k)\}$ do not depend on k. An interesting result, relating stationarity to stability is the following

Theorem 4.1
Suppose $\mathbf{\Phi}(k-1) = \mathbf{\Phi}$, $\mathbf{Q}(k-1) = \mathbf{Q}$ and that $\mathbf{\Phi}$ is asymptotically stable. Then the distribution of $\{\mathbf{x}(k)\}$ as defined in (4.6a) will, as k increases, converge towards a stationary distribution, that is a distribution not depending on k.

The proof, which is beyond the scope of these notes, is given in [1]. In order to find this stationary distribution, note that when stationarity has occurred, $\mathbf{x}(k-1)$ and $\mathbf{x}(k)$ follow the same distribution. In particular, their expectations and covariance matrices must be equal. Denote $\overline{\mu}$ and $\overline{\mathbf{P}}$ the expectation and covariance matrix, respectively, in the stationary distribution. Using (4.6a) and the rules for expectations and variances, we then must have

$$\overline{\mu} =\mathbf{\Phi}\overline{\mu}, \tag{4.8a}$$

$$\overline{\mathbf{P}} =\mathbf{\Phi}\overline{\mathbf{P}}\mathbf{\Phi}^T + \mathbf{Q}. \tag{4.8b}$$

The first equation (involving n unknowns) may be solved directly by recognizing that a non-zero value of $\bar{\mu}$ is only obtainable if $\mathbf{\Phi}$ has 1 as eigenvalue, in which case μ is the corresponding eigenvector(s).

The second equation is more difficult. Taking into account that $\bar{\mathbf{P}}$ is symmetric, there are a total of $n(n+1)/2$ unknown. For small n's we may write this out as a set of linear equations which can be solved by standard techniques. In the more general case, a possible algorithm is to start with an arbitrary symmetric matrix $\bar{\mathbf{P}}$ which is put into the right hand side of the equation, giving a new matrix $\bar{\mathbf{P}}$. This new matrix may again be put into the right hand side to give yet another matrix. Continuing in this manner will in the end give the approeriate matrix.

4.3 Remarks

The general model contains $\{\mathbf{\Phi}(k)\}$,$\{\mathbf{H}(k)\}$,$\{\mathbf{Q}(k)\}$ and $\{\mathbf{R}(k)\}$ as input quantities, in sum a huge number of coefficients. Many of them will in practice be determined by the physical situation and often some of them are the same for all k (the variances for example), but that is not to say that the dependency on k is not important. On the contrary, the fact that the theory goes through for arbitrary parameter sequences is of the highest importance for practical modeling and data processing, as we shall see. Usually, especially in statistics, there are a few unknown parameters that must be found empirically by statistical estimation. We discuss this aspect in Chapter 10. In Chapter 7 we will present a number of specific examples that can be put into the linear state space framework, but a few remarks on modeling is best made in this general set-up.

One issue is initialization. This means that $\mathbf{x}(0)$ must be assigned a value, although not necessarily a fixed one. The standard approach is to assume a probability distribution for $\mathbf{x}(0)$, usually in terms of a

$$E[\mathbf{x}(0)] = \mu(0) \tag{4.9}$$

and a covariance matrix

$$\text{VAR}[\mathbf{x}(0)] = \mathbf{P}(0|0). \tag{4.10}$$

To discuss the meaning of these assumptions, consider Example 1 in Chapter 2 where $x_1(k)$ was the position and $x_2(k)$ the speed at time t_k. We might actually know the position and speed at the start. The right hand side of (4.9) is then set equal to these given values whereas the variance matrix on the right of (4.10) would be

$$\mathbf{P}(0|0) = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$$

Since there is no uncertainty regarding the initial conditions, the variances of the initial state is simply zero. The opposite case would be one where nothing is known (a so-called **diffuse** or **noninformative** prior). Ideally this means that

$$\mathbf{P}(0|0) = \begin{pmatrix} \infty & \cdot \\ \cdot & \infty \end{pmatrix},$$

(the off–diagonal is immaterial), but since infinity can not be represented in a computer, a sufficiently large number will do. There are innumerable intermediate positions between these two extremes. For example, if the position was known perfectly, but the speed was completely in the dark, take

$$\mathbf{P}(0|0) = \begin{pmatrix} 0 & \cdot \\ \cdot & \infty \end{pmatrix},$$

(again the off-diagonal elements do not matter). There is clearly a great deal of flexibility. We shall see in the exercises how the reconstruction of the $\mathbf{x}$-process is influenced by this initial information.

In some situations, we would like to specify $\mathbf{P}(0|0)$ somewhere in between these two extremes. This happens when *some* information is available about $\mathbf{x}(0)$, but not in an exact manner. One possibility is for instance that a similar system have been analyzed before, giving some indication on what $\mathbf{x}(0)$ should be. Alternatively, experts in the field may have knowledge about the system which can be used to make some (necessarily imprecise) statement about the initial state. This would correspond to the *Bayesian* world in statistics, in contrast to the frequentist view in which we put the diagonals to infinity.

Another point is *missing observations*. Gaps in an observation sequence is a quite common phenomenon. Sometimes the measurement equipment is deliberately switched off to be used for other purposes, or the technology involved are occasionally subject to bugs and errors. The most natural way of entering missing information into a model is through infinite variances in the measurement equation. To be specific, consider the observation of position in the second order model, that is $z(k) = x_1(k) + v(k)$. One way to express mathematically that observation $z(k)$ is missing, is to specify an infinite (or very large) variance for the error $v(k)$. This says that the uncertainty is so great that it overshadows the information in $z(k)$, and the effect of $z(k)$ vanishes from the computation. An alternative, and perhaps a more subtle way, is to rewrite the measurement equation as

$$z(k) = H(k)x_1(k) + v(k)$$

where

$$H(k) = \begin{cases} 1, & \text{if } z(k) \text{ is observed;} \\ 0, & \text{if it is not.} \end{cases}$$

Note that $z(k) = v(k)$, if $H(k) = 0$, but now there is no relation to the **x**-process, and $z(k)$ becomes valueless. This second option (which is exact whereas the first one is only approximate) is at least as good as the other. Note that by using these techniques any observational pattern, however complicated, can conveniently be expressed and fed to the computer. The Kalman filter described in Chapter 6 is available to estimate a general **x**-process.

4.4 Problems

Exercise 4.1 (Random walk, extended)
Consider a slight generalization of eq. (4.1) giving the system equation:

$$x(k) = ax(k-1) + w(k-1),$$

where we will assume $w(k-1)$ is Gaussian distributed with expectation equal to zero and variance σ^2.

(a) Write a routine in *Matlab* with input a, σ and N which simulates the system for $k = 1, ..., N$ with $x(0) = 0$.

(Hint: The routine `randn(N)` generates N Gaussian distribution variables with zero expectation and variance equal to 1. By multiplying the variables you get with σ, you obtain new variables with variance equal to σ^2.)

(b) Run the routine in (a) for $N = 1000, \sigma = 1$ and $a = \pm 0.3, \pm 0.9, \pm 1.0$ and ± 1.01, and plot the series. Comment on the results.

(c) Try to relate the results in (b) to the *stabilization* concept in Chapter 2.

(d) Adjust your routine from (a) so that $x(0)$ is drawn from a Gaussian distribution with mean zero and variance $P(0|0)$, where $P(0|0)$ is large (for example equal to 10000). Simulate again the process for different values of a. Comment on how $x(1000)$ depends on the initial state.

Exercise 4.2 (Simulation of two-dimensional systems)
Consider a stochastic system model of the form

$$\begin{pmatrix} x_1(k) \\ x_2(k) \end{pmatrix} = \mathbf{\Phi} \begin{pmatrix} x_1(k-1) \\ x_2(k-1) \end{pmatrix} + \begin{pmatrix} 0 \\ w(k-1) \end{pmatrix}$$

where $\{w(k)\}$ is a white noise process with $w(k-1) \sim \mathcal{N}(0, \sigma^2)$.

(a) Write a *Matlab* routine with input $\mathbf{\Phi}$, σ^2, T and N which simulates the system process for $k = 1, ..., N$. Specify the initial conditions yourself.

(b) Consider now a stochastic version of the (discretized) moving body example (Example 1) from Chapter 2 given by

$$\begin{pmatrix} x_1(k) \\ x_2(k) \end{pmatrix} = \begin{pmatrix} 1 & T \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x_1(k-1) \\ x_2(k-1) \end{pmatrix} + \begin{pmatrix} 0 \\ w(k-1) \end{pmatrix}.$$

Use the routine written in (a) to simulate the process for $T = 0.5$ and $k = 1, ..., 100$. Use different values of σ^2, and plot the results.

(c) Repeat (b) but now with the stochastic version of the (discrete) pendulum movement system in Exercise 2.3, that is $\mathbf{\Phi} = \exp\{\mathbf{A} \cdot T\}$, where

$$\mathbf{A} = \begin{pmatrix} 0 & 1 \\ -g/L & 0 \end{pmatrix}.$$

Use $g = 10.0, L = 1.0$ and $T = 0.5$.

(d) Repeat (b) once more but now including air resistance into the pendulum model (Exercise 2.4), that is $\mathbf{\Phi} = \exp\{\mathbf{A} \cdot T\}$, where now

$$\mathbf{A} = \begin{pmatrix} 0 & 1 \\ -g/L & -c/L \end{pmatrix}.$$

Use again $g = 10.0, L = 1.0, c = 0.5$ and $T = 0.5$.

(e) Comment on the results above. In particular, try to relate the results to the *stabilization* concept in Chapter 2.

(f) From Theorem 4.1, we then know that the systems which are stable will converge to a stationary distribution. Use (4.8) to find the stationary distribution for this (or these) system(s).

(Hint: It is not possible to find $\bar{\mathbf{P}}$ through matrix manipulations. You will have to write down the linear equations for the *components* in $\bar{\mathbf{P}}$ and solve this system.

Chapter 5

Estimation of random variables

Recovering the $\mathbf{x}$-process from measurements is the essential problem for models as described in Section 4.2. This estimation problem differ in an important aspect from the more standard parameter estimation problems considered in introductory courses in statistics. In this case the quantity of interest, $\mathbf{x}(k)$ is a *random* variable, and not a fixed constant as is the case for parameters.

In principle, there are many ways of solving the particular estimation problem involved for the dynamic linear model. One possibility is to base an estimation process on the observation model (4.6b) alone. For instance, if we have the univariate model $n = m = 1$ with $H(k) = 1$, we could estimate $x(k)$ by $\hat{x}(k) = z(k)$. Such an estimate is *unbiased*, since

$$E[\hat{x}(k) - x(k)] = E[z(k) - x(k)] = E[v(k)] = 0,$$

but the variance is as large as that of the original observation. Usually $x(k)$ will be a smooth process, and there is an enormous benefit in utilizing the dynamics in the x-process to aid the reconstruction through feedback, which the former, naive attempt did not do. We discuss how this idea is carried out in Chapter 6 by introducing the Kalman filter that in a certain sense is the best procedure that can be constructed. The Kalman filter applies to linear stochastic state space models and organizes the computations in an elegant recursive manner. We introduce this class of models in the next section. In this chapter we will discuss estimation of random variables in a more general setting, which will give us the basis for deriving the Kalman filter. Section 5.1 consider how to construct optimal estimators in general, while Section 5.2 specialize the results from Section 5.1 to Gaussian distributions.

5.1 Optimal estimation in general.

When deriving the Kalman filter, we will be concerned with estimation of an unobserved random vector $\mathbf{x} \in \mathcal{R}^n$ based on some observed random vector $\mathbf{z} \in \mathcal{R}^m$. Let

$$p(\mathbf{x}, \mathbf{z}) = p(\mathbf{x}|\mathbf{z})p(\mathbf{z}) \tag{5.1}$$

be the joint density of $(\mathbf{x}, \mathbf{z})$, which is factorized into the product of the conditional density of $\mathbf{x}$ given $\mathbf{z}$ and the marginal density of $\mathbf{z}$. Consider the conditional mean (or regressor)

$$\hat{\mathbf{x}} = E[\mathbf{x}|\mathbf{z}] = \int \mathbf{x} p(\mathbf{x}|\mathbf{z}) d\mathbf{x}, \tag{5.2}$$

which (obviously) is a function of $\mathbf{z}$. We shall sometimes write (when we want to emphasize this dependency) $\hat{\mathbf{x}} = \hat{\mathbf{x}}(\mathbf{z})$. The conditional mean has the following important properties:

Theorem 5.1
Assume $\hat{\mathbf{x}} = \hat{\mathbf{x}}(\mathbf{z})$ is given by (5.2). Then the following holds:

(*a*) $E[\hat{\mathbf{x}}(\mathbf{z}) - \mathbf{x}] = \mathbf{0}$.

(*b*) $E[(\hat{\mathbf{x}}(\mathbf{z}) - \mathbf{x})^T \mathbf{g}(\mathbf{z})] = 0$ *for any function* $\mathbf{g}(\mathbf{z}) : \mathcal{R}^m \to \mathcal{R}^n$ *and*
$E[(\hat{\mathbf{x}}(\mathbf{z}) - \mathbf{x})\tilde{\mathbf{g}}(\mathbf{z})^T] = \mathbf{0}$ *for any function* $\tilde{\mathbf{g}}(\mathbf{z}) : \mathcal{R}^m \to \mathcal{R}^p$

(*c*) $E[(\hat{\mathbf{x}}(\mathbf{z}) - \mathbf{x})^T(\hat{\mathbf{x}}(\mathbf{z}) - \mathbf{x})] \leq E[(\mathbf{h}(\mathbf{z}) - \mathbf{x})^T(\mathbf{h}(\mathbf{z}) - \mathbf{x})]$ *for any function* $\mathbf{h}(\mathbf{z}) : \mathcal{R}^m \to \mathcal{R}^n$.

The interpretation from an estimation point of view is immediate. Suppose $\hat{\mathbf{x}}(\mathbf{z})$ is used as an estimate for $\mathbf{x}$. Then (a) implies that the estimate is **unbiased**, i.e. the estimation error is zero in the mean. The error is also *uncorrelated* with the observation vector $\mathbf{z}$ (from (b)), and the average error *is smaller* than for any other alternative method according to (c).

Proof of Theorem 5.1 We will use the double expectation rule. Start by observing that

$$\begin{aligned}
E[(\hat{\mathbf{x}}(\mathbf{z}) - \mathbf{x})^T \mathbf{g}(\mathbf{z})] =& E[E[(\hat{\mathbf{x}}(\mathbf{z}) - \mathbf{x})^T \mathbf{g}(\mathbf{z})|\mathbf{z}]] \\
=& E[(\hat{\mathbf{x}}(\mathbf{z}) - E[\mathbf{x}|\mathbf{z}])^T \mathbf{g}(\mathbf{z})] \\
=& E[(\hat{\mathbf{x}}(\mathbf{z}) - \hat{\mathbf{x}}(\mathbf{z}))^T \mathbf{g}(\mathbf{z})] \\
=& 0,
\end{aligned}$$

and the first part of property (b) of the theorem follows. The second part is shown similarly, only replacing $\mathbf{g}$ with $\tilde{\mathbf{g}}$. (a) is simply a specialization by taking $\tilde{\mathbf{g}}(\mathbf{z}) \equiv 1$. To establish (c) note that

$$\begin{aligned}
&(\mathbf{h}(\mathbf{z}) - \mathbf{x})^T(\mathbf{h}(\mathbf{z}) - \mathbf{x}) \\
&\qquad =(\mathbf{h}(\mathbf{z}) - \hat{\mathbf{x}}(\mathbf{z}))^T(\mathbf{h}(\mathbf{z}) - \hat{\mathbf{x}}(\mathbf{z})) + (\hat{\mathbf{x}}(\mathbf{z}) - \mathbf{x})^T(\hat{\mathbf{x}}(\mathbf{z}) - \mathbf{x}) + \\
&\qquad\quad 2(\hat{\mathbf{x}}(\mathbf{z}) - \mathbf{x})^T(\mathbf{h}(\mathbf{z}) - \hat{\mathbf{x}}(\mathbf{z})).
\end{aligned}$$

Take expectations on both sides. Then the last term on the right vanishes (use $\mathbf{g}(\mathbf{z}) = \mathbf{h}(\mathbf{z}) - \hat{\mathbf{x}}(\mathbf{z})$ in (b)). Hence

$$\begin{aligned}
&E[(\mathbf{h}(\mathbf{z}) - \mathbf{x})^T(\mathbf{h}(\mathbf{z}) - \mathbf{x})] \\
&\qquad = E[(\mathbf{h}(\mathbf{z}) - \hat{\mathbf{x}}(\mathbf{z}))^T(\mathbf{h}(\mathbf{z}) - \hat{\mathbf{x}}(\mathbf{z}))] + E[(\hat{\mathbf{x}}(\mathbf{z}) - \mathbf{x})^T(\hat{\mathbf{x}}(\mathbf{z}) - \mathbf{x})] \\
&\qquad \geq E[(\hat{\mathbf{x}}(\mathbf{z}) - \mathbf{x})^T(\hat{\mathbf{x}}(\mathbf{z}) - \mathbf{x})],
\end{aligned}$$

since the first term on the right of the first line can not be negative. ■

5.2 Optimal estimation in the Gaussian case

Theorem 5.1 provides a strong argument for estimating a random quantity through its conditional expectation given the observations. This may be extremely difficult computationally with the dynamic model in Section 4.2, even with complete knowledge of the statistical properties of the error processes. However, there is one type of situation where the problem simplifies nicely, i.e. when both error processes in the state space model (4.6) are Gaussian. The final results along this line will be proved in the next chapter, but we shall go a long way in the present one by reviewing some basic results about conditional distributions for Gaussian random variables. In Section 3.5 we showed that if $(\mathbf{x}, \mathbf{z})^T$ is Gaussian with mean vector

$$\mu = \begin{pmatrix} \mu_{\mathbf{x}} \\ \mu_{\mathbf{z}} \end{pmatrix}$$

and covariance matrix

$$\mathbf{P} = \begin{pmatrix} \mathbf{P}_{\mathbf{xx}} & \mathbf{P}_{\mathbf{xz}} \\ \mathbf{P}_{\mathbf{zx}} & \mathbf{P}_{\mathbf{zz}} \end{pmatrix},$$

then the distribution of $\mathbf{x}$ given $\mathbf{z}$ is also Gaussian with mean vector

$$\mu_{\mathbf{x}|\mathbf{z}} = \mu_{\mathbf{x}} + \mathbf{P}_{\mathbf{xz}}\mathbf{P}_{\mathbf{zz}}^{-1}(\mathbf{z} - \mu_{\mathbf{z}}) \tag{5.3}$$

and covariance matrix

$$\mathbf{P}_{\mathbf{x}|\mathbf{z}} = \mathbf{P}_{\mathbf{xx}} - \mathbf{P}_{\mathbf{xz}}\mathbf{P}_{\mathbf{zz}}^{-1}\mathbf{P}_{\mathbf{zx}}. \tag{5.4}$$

Both the identities (5.3) and (5.4) are important from an estimation point of view. Suppose $\mathbf{x}$ is some *unknown* vector to be estimated from *observations* $\mathbf{z}$. Then Theorem 5.1, in combination with (5.3) shows that the optimal estimator $\hat{\mathbf{x}} = E[\mathbf{x}|\mathbf{z}] = \mu_{\mathbf{x}|\mathbf{z}}$ is in the Gaussian case a *linear function* of $\mathbf{z}$. Applied to the filtering problem of Section 4.2 this implies that the optimal estimator for $\mathbf{x}(k)$ must be a linear function of $\mathbf{z}(1), ..., \mathbf{z}(k)$ *when the noise processes in* (4.6) *both are Gaussian.* The other identity reveals the remarkable property that the conditional covariance *is independent of the given observations* $\mathbf{z}$. Consider this in the estimation context. We know that

$$\text{VAR}[\mathbf{x} - \mu_{\mathbf{x}|\mathbf{z}}|\mathbf{z}] = E[(\mathbf{x} - \mu_{\mathbf{x}|\mathbf{z}})(\mathbf{x} - \mu_{\mathbf{x}|\mathbf{z}})^T|\mathbf{z}] = \mathbf{P}_{\mathbf{x}|\mathbf{z}}$$

by definition. Note that $-(\mathbf{x} - \mu_{\mathbf{x}|\mathbf{z}})$ coincides with the estimation error

$$\mathbf{e} = \hat{\mathbf{x}} - \mathbf{x}.$$

Hence

$$E[\mathbf{e}\mathbf{e}^T|\mathbf{z}] = \mathbf{P}_{\mathbf{x}|\mathbf{z}}.$$

Then, from the double expectation rule (Theorem 3.1)

$$E[\mathbf{e}\mathbf{e}^T] = E[E[\mathbf{e}\mathbf{e}^T|\mathbf{z}]] = E[\mathbf{P}_{\mathbf{x}|\mathbf{z}}] = \mathbf{P}_{\mathbf{x}|\mathbf{z}}. \tag{5.5}$$

Since, by property (a) of Theorem 5.1 $\mathbf{e}_1$ has expectation 0, this is equal to the covariance matrix for $\mathbf{e}_1$,

$$\text{VAR}[\mathbf{e}_1] = \mathbf{P}_{1|2},$$

and we have in the realm of Gaussian models a particularly simple expression for the (unconditional) estimation errors, which can be computed ahead of the actual data, as can the conditional one.

Note that although for our *linear* models, the unconditional variances of the estimation errors coincides with the conditional ones, this is not a general result. Furthermore, their *interpretations* are different. The conditional variance reflects the uncertainty involved for *those particular data that we have observed*, while the unconditional variance defines the uncertainty involved if we were able to repeat our experiments several times.

5.3 Problems

Exercise 5.1 (Gaussian estimation)
Let θ be an unknown quantity. Assume there exists one single measurement y where

$$y = h\theta + v.$$

Here h is a parameter known from the measurement technology while v is a noise term which we assume is Gaussian distributed with expectation 0 and variance R. Also assume that θ is a Gaussian random variable with expectation μ_θ and variance P_θ.

(a) We will assume in the whole exercise that there is no dependence between θ and v. Discuss the validity of this assumption.

(b) Show that y is Gaussian distributed with expectation $\mu_y = h\mu_\theta$ and variance $P_y = h^2 P_\theta + R$.

(c) Calculate $\text{cov}[\theta, y]$. Specify also $\text{corr}[\theta, y]$.

(d) Show that the conditional distribution for θ given y is Gaussian with expectation

$$E[\theta|y] = \mu_\theta + \frac{hP_\theta}{h^2 P_\theta + R}(y - h\mu_\theta)$$

and variance

$$\text{var}[\theta|y] = P_\theta \frac{R}{h^2 P_\theta + R}.$$

(e) Write down the “best” way of estimating θ based on y.

We will in the following interpret the expressions for $E[\theta|y]$ and $\text{var}[\theta|y]$ based on the estimation problem defined in the beginning of the exercise.

(f) Analyze how the expressions depend on P_θ. Try to explain the structure of this dependence.

(g) What happens when $P_\theta \to \infty$? Try to explain which result this is a special case of.

(h) How do the expressions for $E[\theta|y]$ and $\text{var}[\theta|y]$ depend on R? Explain.

(i) What happens with the two expressions when $R = 0$? Explain again.

(j) And how is the dependence on h? Explain once more.

Exercise 5.2 (Prior information from earlier experiments)
In Exercise 5.1 it was assumed that some knowledge about θ was available *before* the measurement y was collected (we knew the expectation and variance of θ). Assume now that we do not know anything about θ. Then we get to know the following: A measurement $z = \theta + u$, where u is Gaussian noise with expectation 0 and variance P_θ, has resulted in the value $z = \mu_\theta$. Explain why we after measuring z (but before measuring y) have exactly the same type of a priori knowledge about θ as in Exercise 5.1. (A priori because we have no realizations of y yet.)

Exercise 5.3 (A simple state space model (AR(1)) – part I)
Consider the model $x(k+1) = ax(k) + w(k)$ for $k = 1, \ldots, N$. Here $x(1) \sim \mathcal{N}(0, \sigma^2)$. The random variables $w(1), w(2), \ldots, w(N)$ are independent and identical distributed $\mathcal{N}(0, Q)$. They are also independent of $x(1)$. The constant a is assumed to be in the interval $(-1, 1)$.

(*a*) Explain why $x(k)$ for all $k = 1, \dots, N$ are Gaussian distributed and show that $\mu(k) = E[x(k)] = 0$.

(*b*) Derive a set of difference equations for

$$p_x(k) = \text{var}[x(k)] = E[(x(k) - \mu(k))^2] = E[x(k)^2]$$

by taking the variance on each side of the equality sign in the model. Show that we get $p_x(k+1) = a^2 p_x(k) + Q$. Why is this deterministic difference equation stable?

(*c*) When $k \to \infty$, then $p_x(k)$ will converge towards a limit $\bar{p}_x$. Show that $\bar{p}_x = Q/(1 - a^2)$.

(*d*) Assume $\sigma = 1$ and $Q = 4$. Use *Matlab* for plotting $p_x(k)$ for all $k = 1, \dots, 20$ for $a = -0.7$ and $a = 0.4$. Try to make some conclusions on your results.

(*e*) Let $a = -0.7$ and make a large number of realizations of the process. Plot the *empirical* distributions at time points $k = 1$, $k = 3$ and $k = 15$ in *Matlab* . In the *same* plots, place the theoretical Gaussian distributions for $x(1)$, $x(3)$ and $x(15)$. The similarity should be good. Be sure you understand why this comparison makes any sense.

(Hint: The *Matlab* routine `hist` draws a histogram (empirical distribution) of a dataset. The `hold` routine may be useful in making several figures in the same plot.)

Exercise 5.4 (A simple state space model (AR(1)) – part II)
We will consider the same model as in Exercise 5.3, but now use $Q = \sigma^2(1 - a^2)$.

(*a*) Explain why all $x(k)$ now have the same distribution. Make a plot of the distribution for $\sigma = 1$.

Which values would you use for estimating an unknown $x(k)$?

(*b*) Find the conditional distribution for $x(k)$ given $x(k-1)$. Make a plot also for this distribution and compare with the one you found in (*a*).

Let $a = -0.7$ and assume $x(k-1) = 0.5$. How would you now estimate $x(k)$?

(*c*) Show that the conditional distribution for $(x(k), x(k+1))$ given $x(k-1)$ is a Gaussian distribution with expectation vector

$$\begin{pmatrix} a \\ a^2 \end{pmatrix} x(k-1)$$

and covariance matrix

$$\begin{pmatrix} 1 & a \\ a & 1 + a^2 \end{pmatrix} (1 - a^2)\sigma^2.$$

Make a plot of the *joint* conditional distribution for $(x(k), x(k+1))$ given $x(k-1)$.

How would you estimate $(x(k), x(k+1))$ if $x(k-1)$ was known?

(Hint: The routines `meshgrid` and `mesh` may be useful.)

(d) Find the joint *unconditional* distribution for $(x(k), x(k+1))$. Make a plot of the distribution and compare with the one you made in (c).

How would you estimate $(x(k), x(k+1))$ if $x(k-1)$ was unknown?

(e) Use the result from (c) to find the joint distribution for $(x(k-1), x(k), x(k+1))$.

(f) What is the conditional distribution for $x(k)$ given $(x(k-1), x(k+1))$? Try to explain why this conditional distribution is equal to the conditional distribution for $x(k)$ given $x(1), \ldots, x(k-1), x(k+1), \ldots, x(N)$.

Chapter 6

The discrete time Kalman filter

Our objective in this chapter is to construct estimators $\hat{\mathbf{x}}(k)$ for $\mathbf{x}(k)$ at time k with error estimates. This is usually called **state estimation**. Since we are assuming that each observation $\mathbf{z}(k)$ is becoming known at exactly time instant k (or rather some t_k), the data available for the estimation of $\mathbf{x}(k)$ is $\mathbf{z}(1), ..., \mathbf{z}(k)$. The Kalman filter provides the best estimate of $\mathbf{x}(k)$ that can be obtained through *linear* estimators (6.1), i.e. it is of the form

$$\hat{\mathbf{x}}(k) = \mathbf{a}_k(k)\mathbf{z}(k) + \mathbf{a}_k(k-1)\mathbf{z}(k-1) + ... + \mathbf{a}_k(1)\mathbf{z}(1), \tag{6.1}$$

where $\{\mathbf{a}_k(l)\}$ are coefficients to be adapted from knowledge of the model and some initial information about $\mathbf{x}(0)$, the start vector of the system process $\{\mathbf{x}(k)\}$. The coefficients will obviously vary from one point in time to another, explaining the index k on the $\mathbf{a}$'s.

The **estimation error** when $\mathbf{x}(k)$ is estimated by $\hat{\mathbf{x}}(k)$ is

$$\mathbf{e}(k) = \hat{\mathbf{x}}(k) - \mathbf{x}(k). \tag{6.2}$$

The uncertainty of the estimation error is of vital interest when applying the Kalman filter. We will see that uncertainty measures based on variances are to be computed as part of the scheme.

There are many derivations of the Kalman filter in the literature. Often *ad hoc* assumptions (such as linearity of the filter) are introduced *a priori* to simplify the computations. The derivation given here has the advantage of being rigorous and self–contained in addition to give a useful background in general estimation theory.

Throughout the discussion, we will for simplicity assume that the input sequence $\mathbf{u}(k) = 0$ for all k. Non–zero input functions can easily be incorporated, but would only make the derivations to follow more complex without gaining any extra insight into the main ideas. The extension of the Kalman filter to non-zero input-vectors will therefore only be presented without proof at the end of the chapter.

Throughout this chapter the sequences $\{\mathbf{\Phi}(k)\}$, $\{\mathbf{H}(k)\}$, $\{\mathbf{Q}(k)\}$ and $\{\mathbf{R}(k)\}$ are regarded as fully known. The Kalman filter is a *linear* estimation method.

Some notation and concepts will be introduced in Section 6.1. The Kalman filter will then be derived under Gaussian conditions in Section 6.2. In Section 6.4 we illustrate the use of the Kalman filter through a few examples. Section 6.5 demonstrates by a very simple argument that the Kalman filter actually also obtains the best *linear* estimate in the general non-Gaussian case. Finally in Section 6.6 we will generalize the filter for non-zero deterministic inputs.

6.1 Preliminaries

Go back to the problem of Section 4.2, i.e. that of estimating $\mathbf{x}(k)$ as accurately as possible on the basis of

$$\vec{\mathbf{z}}(l) = \begin{pmatrix} \mathbf{z}(1) \\ \mathbf{z}(2) \\ \vdots \\ \mathbf{z}(l) \end{pmatrix}. \tag{6.3}$$

Define

$$\hat{\mathbf{x}}(k|l) = E[\mathbf{x}(k)|\vec{\mathbf{z}}(l)] \tag{6.4}$$

for general k and l. The solution to the estimation problem above is given by Theorem 5.1 as the conditional mean $\hat{\mathbf{x}}(k|k) = E[\mathbf{x}(k)|\vec{\mathbf{z}}(k)]$. This quantity can in the Gaussian case, at least in principle, be computed through the results of the preceding chapter. The point is that when both error processes $\{\mathbf{w}(k)\}$ and $\{\mathbf{v}(k)\}$ in (4.6) are Gaussian, then, according to Theorem 3.2, so is $\mathbf{x}(k)$ and $\vec{\mathbf{z}}(k)$ (and jointly too), and the formula for conditional expectations (5.3) yields $\hat{\mathbf{x}}(k|k)$ after the identification of the appropriate cross covariance matrices. In practice this is not the way we do it, a major drawback being that the whole computation has to be done all over again for each single k. Even worse, the approach requires the inversion of the covariance matrix of $\vec{\mathbf{z}}(k)$ (corresponding to $\mathbf{P}_{22}$ in (5.3)), which turns into a huge matrix as the time is running and k becomes larger and larger. Surely we should seek some recursion that refers as much as possible of the computations at k back to what has already been done at $k-1$ and earlier. This is exactly what is accomplished by the Kalman filter, although the ensuing derivation will show that it is really more than merely a convenient computational tool.

Introduce first some more mathematical notation. We shall need the auxiliary conditional mean

$$\hat{\mathbf{x}}(k|k-1) = E[\mathbf{x}(k)|\vec{\mathbf{z}}(k-1)] \tag{6.5}$$

which is the predictor using information up to $k-1$ only. (Estimation of future random variables is usually denoted *prediction*.) The estimation error when estimating $\mathbf{x}(k)$ by $\hat{\mathbf{x}}(k|l)$ is

$$\mathbf{e}(k|l) = \hat{\mathbf{x}}(k|l) - \mathbf{x}(k). \tag{6.6}$$

The variability of this term is important in order to evaluate the quality of the estimate. A popular variability measure is the covariance matrix. Recall from Theorem 5.1(a) that $\hat{\mathbf{x}}(k|l) - \mathbf{x}(k)$ has zero mean. Hence the error has covariance matrix

$$\mathbf{P}(k|l) = \text{VAR}[\mathbf{e}(k|l)] = E[(\hat{\mathbf{x}}(k|l) - \mathbf{x}(k))(\hat{\mathbf{x}}(k|l) - \mathbf{x}(k))^T]. \tag{6.7}$$

This covariance matrix measures the variability we will expect using $\hat{\mathbf{x}}(k|l)$ without taking the actual values of $\vec{\mathbf{z}}(l)$ into account, that is a measure of performance of a *procedure*. When the estimation is to be performed at time l, the observations $\vec{\mathbf{z}}(l)$ are available, and in this case it is more natural to consider the *conditional* covariance matrix $\text{VAR}[\mathbf{e}(k|l)|\vec{\mathbf{z}}(l)]$. In general, $\text{VAR}[\mathbf{e}(k|l)|\vec{\mathbf{z}}(l)] \neq \text{VAR}[\mathbf{e}(k|l)$. However, as shown in Section 5.2, these covariance matrices coincide under Gaussian assumptions.

For the derivation of the Kalman filter, we will in particular be interested in the situation where $l = k$, while the case $l = k-1$ will be important as part of the algorithm. For future prediction the case $l < k$ in general will be of interest and is discussed in Section 9.1. In Chapter 9, considering so-called smoothing, also the case $l > k$ will be treated.

Another important concept, is the so-called **innovation process**, defined through

$$\tilde{\mathbf{z}}(k|k-1) = \mathbf{z}(k) - \mathbf{H}(k)\hat{\mathbf{x}}(k|k-1). \tag{6.8}$$

To motivate the name recall from (4.6b) that $E[\mathbf{z}(k)|\mathbf{x}(k)] = \mathbf{H}(k)\mathbf{x}(k)$ so that the term $\mathbf{H}(k)\hat{\mathbf{x}}(k|k-1)$ would be our natural guess for $\mathbf{z}(k)$ before it was actually observed, i.e. at time $k-1$. The difference $\mathbf{z}(k) - \mathbf{H}(k)\hat{\mathbf{x}}(k|k-1)$ may thus be interpreted as *the unexpected part* of the observation $\mathbf{z}(k)$, that is an *innovation* we were not able to foresee one time unit earlier.

The innovation process $\{\tilde{\mathbf{z}}(k|k-1)\}$ plays a central role, not only for the Kalman filtering theory to be derived here, but for linear filtering in general. Lemma 6.1 summarizes its main properties.

Lemma 6.1
The innovation sequence defined by (6.8) *has the following properties:*

$$E[\tilde{\mathbf{z}}(k|k-1)] = \mathbf{0}, \tag{6.9a}$$

$$COV[\tilde{\mathbf{z}}(k|k-1), \mathbf{z}(k')] = \mathbf{0} \qquad \textit{for all } k' < k, \tag{6.9b}$$

$$\mathbf{S}(k) = VAR[\tilde{\mathbf{z}}(k|k-1)] = \mathbf{H}(k)\mathbf{P}(k|k-1)\mathbf{H}(k)^T + \mathbf{R}(k). \tag{6.9c}$$

where $\mathbf{R}(k)$ *is the covariance matrix of* $\mathbf{v}(k)$, *cf.* (4.7b).

The innovation at time k has thus mean zero and *is uncorrelated with all observations prior to k* (property (6.9b)). The latter observation is extremely important. It will in the present sequel enable us to simplify the argument leading to the Kalman filter. Later (in Chapter 10) we shall see how it yields a neat decomposition of the joint density functions of the observations.

Proof of Lemma 6.1 Start by inserting (4.6b) for $\mathbf{z}(k)$ into (6.8). Then

$$\tilde{\mathbf{z}}(k|k-1) = \mathbf{H}(k)(\mathbf{x}(k) - \hat{\mathbf{x}}(k|k-1)) + \mathbf{v}(k), \tag{6.10}$$

and after replacing $\hat{\mathbf{x}}(k|k-1)$ by (6.5)

$$\tilde{\mathbf{z}}(k|k-1) = \mathbf{H}(k)\{\mathbf{x}(k) - E[\mathbf{x}(k)|\vec{\mathbf{z}}(k-1)]\} + \mathbf{v}(k). \tag{6.11}$$

We know from Theorem 5.1(a) that $\mathbf{x}(k)-E[\mathbf{x}(k)|\vec{\mathbf{z}}(k-1)]$ has mean zero. This yields property (6.9a) of the lemma, since $E[\mathbf{v}(k)] = 0$. To establish (6.9b), observe that the first term on the right of (6.11) is uncorrelated with $\vec{\mathbf{z}}(k-1)$ (by Theorem 5.1(b)) as is also the second term $\mathbf{v}(k)$. Hence the sum must be uncorrelated too, which is the statement to be proved. The formula for VAR$[\tilde{\mathbf{z}}(k|k-1)]$ is almost immediate. Indeed, the two terms in (6.10) are by model assumptions uncorrelated. Hence

$$\begin{aligned}\mathrm{VAR}[\tilde{\mathbf{z}}(k|k-1)] =& \mathrm{VAR}[\mathbf{H}(k)(\mathbf{x}(k) - \hat{\mathbf{x}}(k|k-1))] + \mathrm{VAR}[\mathbf{v}(k)]\\ =& \mathbf{H}(k)\mathrm{VAR}[\mathbf{x}(k) - \hat{\mathbf{x}}(k|k-1)]\mathbf{H}(k)^T + \mathrm{VAR}[\mathbf{v}(k)]\\ =& \mathbf{H}(k)\mathbf{P}(k|k-1)\mathbf{H}(k)^T + \mathbf{R}(k).\end{aligned}$$

Here the second line is a consequence of the rules for operating covariance matrices, as given by Table 3.3, and the third line follows after recalling (6.7) and (4.7b). ■

6.2 The Kalman filter under Gaussian conditions

The main structure of the Kalman filter is the double recursion

$$\begin{aligned}\hat{\mathbf{x}}(k|k-1) &= \mathbf{\Phi}(k-1)\hat{\mathbf{x}}(k-1|k-1) && \text{(prediction)} \quad (6.12a)\\ \hat{\mathbf{x}}(k|k) &= \hat{\mathbf{x}}(k|k-1) + \mathbf{K}(k)(\mathbf{z}(k) - \mathbf{H}(k)\hat{\mathbf{x}}(k|k-1)) && \text{(updating),} \quad (6.12b)\end{aligned}$$

where $\mathbf{K}(k)$ is a *deterministic* weight matrix known as the **Kalman gain**. Note that the second term in (6.12b) equals $\mathbf{K}(k)\tilde{\mathbf{z}}(k|k-1)$, where $\tilde{\mathbf{z}}(k|k-1)$ is the innovation defined in (6.8). Thus the updating of $\mathbf{x}(k)$ when $\mathbf{z}(k)$ becomes available, amounts to modifying the former estimate $\mathbf{x}(k|k-1)$ by a quantity proportional to the innovative part of the new observation.

The computation of the Kalman gain $\mathbf{K}(k)$ is done jointly with the error covariance matrices $\mathbf{P}(k|k)$ and $\mathbf{P}(k|k-1)$ in one single recursion, also involving $\mathbf{S}(k)$, the covariance matrix for the innovation, as defined in (6.9c) which is of interest in its own right (see Chapter 10). The most common version of the algorithm, the covariance filter, is as follows:

$$\begin{aligned}\mathbf{P}(k|k-1) &= \mathbf{\Phi}(k-1)\mathbf{P}(k-1|k-1)\mathbf{\Phi}(k-1)^T + \mathbf{Q}(k-1) && (6.13a)\\ \mathbf{S}(k) &= \mathbf{H}(k)\mathbf{P}(k|k-1)\mathbf{H}(k)^T + \mathbf{R}(k) && (6.13b)\\ \mathbf{K}(k) &= \mathbf{P}(k|k-1)\mathbf{H}(k)^T\mathbf{S}(k)^{-1} && (6.13c)\\ \mathbf{P}(k|k) &= (\mathbf{I}_n - \mathbf{K}(k)\mathbf{H}(k))\mathbf{P}(k|k-1). && (6.13d)\end{aligned}$$

Clearly this *is* a recursion, since $\mathbf{P}(k-1|k-1)$ at the right of the first equation is transferred to an expression for $\mathbf{P}(k|k)$ in (6.13d). In the literature the scheme is often written with fewer equations. This has no advantages and, moreover, all the quantities computed in (6.13) have a separate role to play.

In (6.13c) the inversion of $\mathbf{S}(k)$ is required. A sufficient condition for this inverse to exist is that $\mathbf{R}(k)$ is positive definite, which we will normally assume. In cases where this condition not is satisfied, some more care has to be taken, although the inverse might exist even in many of these cases.

Several other algebraically equivalent versions of the recursion (6.13) exist (cf. Chapter 8). These are of interest because they may provide additional insight or have improved numerical properties.

Also note that the observations do not go into this second part of the Kalman algorithm at all. In fact, we practically foresaw that earlier after having derived the formula (5.4) for the covariance matrix of conditional Gaussian distributions. The deeper reason for the data independence can be sought in that result. The property has great practical interest for on-line applications, since it means that the Kalman gains, the hardest part of the algorithm, can be computed and stored prior to the actual data processing.

6.3 Derivation of the filter

We will prove the equations (6.12) and (6.13) through four steps.

Step 1 Consider first the prediction step (6.12a) and the prediction error covariance matrix (6.13a). Clearly

$$\begin{aligned}\hat{\mathbf{x}}(k|k-1) &\overset{(6.5)}{=} E[\mathbf{x}(k)|\vec{\mathbf{z}}(k-1)]\\ &\overset{(4.6a)}{=} E[\mathbf{\Phi}(k-1)\mathbf{x}(k-1)|\vec{\mathbf{z}}(k-1)] + E[\mathbf{w}(k-1)|\vec{\mathbf{z}}(k-1)]\\ &= \mathbf{\Phi}(k-1)E[\mathbf{x}(k-1)|\vec{\mathbf{z}}(k-1)] + E[\mathbf{w}(k-1)]\\ &= \mathbf{\Phi}(k-1)\hat{\mathbf{x}}(k-1|k-1) + \mathbf{0},\end{aligned}$$

which is (6.12a). A central ingredient of the argument is that $\mathbf{w}(k-1)$ is uncorrelated with everything going on before k, in particular $\vec{\mathbf{z}}(k-1)$. Similar reasoning yields (6.13a). Start out with

$$\hat{\mathbf{x}}(k|k-1) - \mathbf{x}(k) = \mathbf{\Phi}(k-1)(\hat{\mathbf{x}}(k-1|k-1) - \mathbf{x}(k-1)) - \mathbf{w}(k-1).$$

Take covariance matrices on both sides, recalling that $\mathbf{P}(k|k-1) = \mathrm{VAR}[\hat{\mathbf{x}}(k|k-1) - \mathbf{x}(k)]$. Then

$$\begin{aligned}
&\mathbf{P}(k|k-1)\\
&\quad= \mathrm{VAR}[\mathbf{\Phi}(k-1)(\hat{\mathbf{x}}(k-1|k-1) - \mathbf{x}(k-1)) - \mathbf{w}(k-1)]\\
&\quad= \mathbf{\Phi}(k-1)\mathrm{VAR}[\hat{\mathbf{x}}(k-1|k-1) - \mathbf{x}(k-1)]\mathbf{\Phi}(k-1)^T + \mathrm{VAR}[\mathbf{w}(k-1)]\\
&\quad= \mathbf{\Phi}(k-1)\mathbf{P}(k-1|k-1)\mathbf{\Phi}(k-1)^T + \mathbf{Q}(k-1),
\end{aligned}$$

which is (6.13a).

Step 2 Note that (6.13b) coincides with property (6.9c) of Lemma 6.1, and it remains to prove (6.12b), (6.13c) and (6.13d)). The argument relies on the formula (3.22a) for Gaussian conditional expectations. Observe first that $\vec{\mathbf{z}}(k)$ and $(\vec{\mathbf{z}}(k-1), \tilde{\mathbf{z}}(k|k-1))$ mutually determine each other; see (6.8). Thus $\hat{\mathbf{x}}(k|k) = E[\mathbf{x}(k)|\vec{\mathbf{z}}(k)]$ can also be written

$$\hat{\mathbf{x}}(k|k) = E[\mathbf{x}(k)|\vec{\mathbf{z}}(k-1), \tilde{\mathbf{z}}(k|k-1)]. \tag{6.14}$$

Consider the random vector

$$\begin{pmatrix} \mathbf{x}(k) \\ \vec{\mathbf{z}}(k-1) \\ \tilde{\mathbf{z}}(k|k-1) \end{pmatrix}$$

with mean vector

$$\begin{pmatrix} \mu_{\mathbf{x}}(k) \\ \mu_{\vec{\mathbf{z}}}(k-1) \\ \mathbf{0} \end{pmatrix}$$

and covariance matrix

$$\begin{pmatrix} \mathbf{P}_{\mathbf{x}\mathbf{x}}(k) & \mathbf{P}_{\mathbf{x}\vec{\mathbf{z}}}(k) & \mathbf{P}_{\mathbf{x}\tilde{\mathbf{z}}}(k) \\ \mathbf{P}_{\vec{\mathbf{z}}\mathbf{x}}(k) & \mathbf{P}_{\vec{\mathbf{z}}\vec{\mathbf{z}}}(k) & \mathbf{0} \\ \mathbf{P}_{\tilde{\mathbf{z}}\mathbf{x}}(k) & \mathbf{0} & \mathbf{P}_{\tilde{\mathbf{z}}\tilde{\mathbf{z}}}(k) \end{pmatrix}.$$

Here $\mu_{\mathbf{x}}(k)$ and $\mu_{\vec{z}}(k-1)$ are the means of $\mathbf{x}(k)$ and $\vec{\mathbf{z}}(k-1)$ respectively while $\mathbf{P}_{\mathbf{x}\mathbf{x}}(k)$ is the covariance matrix of $\mathbf{x}(k)$, $\mathbf{P}_{\mathbf{x}\vec{\mathbf{z}}}(k)$ the cross covariance matrix between $\mathbf{x}(k)$ and $\vec{\mathbf{z}}(k-1)$ and so on. Note the zero mean for the innovation process $\tilde{\mathbf{z}}(k|k-1)$

and the zero cross covariance between $\tilde{\mathbf{z}}(k|k-1)$ and $\vec{\mathbf{z}}(k-1)$ are consequences of Lemma 6.1. Apply Corollary 3.1 to the random vector with $\mathbf{x}_1 = \mathbf{x}(k), \mathbf{x}_2 = \vec{\mathbf{z}}(k-1)$ and $\mathbf{x}_3 = \tilde{\mathbf{z}}(k|k-1)$ to see that

$$\begin{aligned}\hat{\mathbf{x}}(k|k) =& E[\mathbf{x}(k)|\vec{\mathbf{z}}(k-1)] + \mathbf{P}_{\mathbf{x}\tilde{\mathbf{z}}}(k)\mathbf{P}^{-1}_{\tilde{\mathbf{z}}\tilde{\mathbf{z}}}(k)\tilde{\mathbf{z}}(k|k-1)\\ =& E[\mathbf{x}(k)|\vec{\mathbf{z}}(k-1)] + \mathbf{K}(k)\tilde{\mathbf{z}}(k|k-1),\end{aligned} \tag{6.15}$$

But we have now arrived at (6.12b) with

$$\mathbf{K}(k) = \mathbf{P}_{\mathbf{x}\tilde{\mathbf{z}}}(k)\mathbf{P}^{-1}_{\tilde{\mathbf{z}}\tilde{\mathbf{z}}}(k). \tag{6.16}$$

Step 3 The next step is to find a suitable expression for $\mathbf{P}(k|k)$, defined in (6.7). From (5.5) it is clear that $\mathbf{P}(k|k) = \mathrm{VAR}[\mathbf{x}(k)|\vec{\mathbf{z}}(k)] = \mathrm{VAR}[\mathbf{x}(k)|\vec{\mathbf{z}}(k-1), \tilde{\mathbf{z}}(k|k-1)]$. Again we can apply Corollary 3.1 to see that

$$\mathbf{P}(k|k) = \mathbf{P}(k|k-1) - \mathbf{P}_{\mathbf{x}\tilde{\mathbf{z}}}(k)\mathbf{P}_{\tilde{\mathbf{z}}\tilde{\mathbf{z}}}(k)^{-1}\mathbf{P}_{\tilde{\mathbf{z}}\mathbf{x}}(k). \tag{6.17}$$

Step 4 For the termination of the argument we have to derive $\mathbf{P}_{\mathbf{x}\tilde{\mathbf{z}}}(k)$, $\mathbf{P}_{\tilde{\mathbf{z}}\mathbf{x}}(k)$ and $\mathbf{P}_{\tilde{\mathbf{z}}\tilde{\mathbf{z}}}(k)$ in (6.16) and (6.17). The latter has also been called $\mathbf{S}(k)$ and is taken care of already. For $\mathbf{P}_{\mathbf{x}\tilde{\mathbf{z}}}(k)$:

$$\begin{aligned}\mathbf{P}_{\mathbf{x}\tilde{\mathbf{z}}}(k) =& \mathrm{COV}[\mathbf{x}(k), \tilde{\mathbf{z}}(k|k-1)]\\ =& E[\mathbf{x}(k)\tilde{\mathbf{z}}(k|k-1)^T]\\ \overset{(6.10)}{=}& E[\mathbf{x}(k)(\mathbf{H}(k)(\mathbf{x}(k) - \hat{\mathbf{x}}(k|k-1)) + \mathbf{v}(k))^T]\\ =& (E[\mathbf{x}(k)(\mathbf{x}(k) - \hat{\mathbf{x}}(k|k-1))^T])\mathbf{H}(k)^T + E[\mathbf{x}(k)\mathbf{v}(k)^T]\\ =& \{E[(\mathbf{x}(k) - \hat{\mathbf{x}}(k|k-1))(\mathbf{x}(k) - \hat{\mathbf{x}}(k|k-1))^T]\\ &+ E[\hat{\mathbf{x}}(k|k-1)(\mathbf{x}(k) - \hat{\mathbf{x}}(k|k-1))^T]\}\mathbf{H}(k)^T + E[\mathbf{x}(k)\mathbf{v}(k)]\\ \overset{(6.7)}{=}& [\mathbf{P}(k|k-1) + \mathbf{0}]\mathbf{H}(k)^T + \mathbf{0},\end{aligned} \tag{6.18}$$

where the first zero is a consequence of the prediction error $\hat{\mathbf{x}}(k|k-1) - \mathbf{x}(k)$ being uncorrelated with any function of $\vec{\mathbf{z}}(k-1)$ (Theorem 5.1(b)), $\hat{\mathbf{x}}(k|k-1)$ in particular, and the second zero is by model assumption directly. When the identities $\mathbf{P}_{\mathbf{x}\tilde{\mathbf{z}}}(k) = \mathbf{P}(k|k-1)\mathbf{H}(k)^T$, $\mathbf{P}_{\tilde{\mathbf{z}}\tilde{\mathbf{z}}}(k) = \mathbf{S}(k)$ and $\mathbf{P}_{\tilde{\mathbf{z}}\mathbf{x}}(k) = \mathbf{P}^T_{\mathbf{x}\tilde{\mathbf{z}}}$ are inserted into (6.16) and (6.17), (6.13c) and (6.13d) follow.

6.4 Examples

In a few situations, explicit analytical expressions can be obtained from the Kalman filter

Example 4 (iid variables)
Consider the simple (but important) example

$$x(k) = x(k-1) \tag{6.19a}$$
$$z(k) = x(k) + v(k) \tag{6.19b}$$

where $v(k) \sim \mathcal{N}(0, \sigma^2)$. This actually corresponds to a situation where, for given $x(k)$, all $z(k)$ are independent and identically distributed (iid) $\mathcal{N}(\mu, \sigma^2)$ where $\mu = x(k)$. Note that there is no noise in the system equation in this case. Consider now first the updating equations for the covariance matrix $P(k|k)$. By (6.13a),

$$P(k|k-1) = P(k-1|k-1)$$

which should be obvious because of (6.19a). Further,

$$S(k) = P(k-1|k-1) + \sigma^2,$$
$$K(k) = \frac{P(k-1|k-1)}{P(k-1|k-1) + \sigma^2},$$

and

$$P(k|k) = \frac{P(k-1|k-1)\sigma^2}{P(k-1|k-1) + \sigma^2},$$

where the last equation gives a recursive equation for calculating $P(k|k)$. We may rewrite the equation as

$$\frac{\sigma^2}{P(k|k)} = 1 + \frac{\sigma^2}{P(k-1|k-1)}$$

which shows that $P(k|k) < P(k-1|k-1)$. Thus, the error variance decreases monotonically as each new observation is processed.

Now assume $P(0|0) = \infty$, corresponding to no prior information about μ. Then it is easy to see that

$$\frac{\sigma^2}{P(k|k)} = k \text{ and } P(k|k) = \frac{\sigma^2}{k}. \tag{6.20}$$

Further, the Kalman gain is in this case

$$K(k) = \frac{\frac{\sigma^2}{k-1}}{\frac{\sigma^2}{k-1} + \sigma^2} = \frac{1}{k}.$$

giving that

$$\begin{aligned}\hat{x}(k|k) =& \hat{x}(k-1|k-1) + \frac{1}{k}(z(k) - \hat{x}(k-1|k-1)) \\ =& \frac{k-1}{k}\hat{x}(k-1|k-1) + \frac{1}{k}z(k)\end{aligned} \tag{6.21}$$

For $k = 1$, we obviously get $\hat{x}(1|1) = z(1)$, and by induction it follows that in general

$$\hat{x}(k|k) = \frac{1}{k}\sum_{i=1}^{k} z(i) = \bar{z}(k). \tag{6.22}$$

Now, of course, in this simple situation we would normally not go through the whole Kalman filter framework, but rather work on the problem more directly (as is done in elementary statistics courses). An interesting consequence of applying the Kalman filter to this problem is however equation (6.21) which gives a recursive algorithm for calculating the mean. Such a recursive calculation is more numerical robust than the more direct method of first summing the variables and then dividing by k. In Chapter 10 we will consider methods for estimating the variance σ^2 and also in this case we will see that a recursive equation appears. □

Here the Kalman filter is seen to provide the classical solution to a well known problem. The Bayesian version is also easily accommodated. We will consider this issue in one of the exercises

The above example can actually be considered as a special case of linear regression which also can be formulated as a stochastic state space model, and for which also analytical expressions for the estimates exist. We will consider this further in Chapter 7.

The situations for which simple analytical expressions are available are in general rather the exception than the rule. Even for one-dimensional examples the calculations quickly become intractable to carry out by hand.

Example 3 (random walk, cont.)
Consider the (small) generalization of Example 4 where we now add noise to the system equation giving the new model

$$x(k) = x(k-1) + w(k-1), \tag{6.23a}$$
$$z(k) = x(k) + v(k). \tag{6.23b}$$

This is the random walk model considered in Section 4.1. In this case analytical expressions for $\hat{x}(k|k)$ and $P(k|k)$ become rather messy. Still however, their values are easily calculated by programming the Kalman filter equations into the computer. Assume, as before, that $Q(k-1) = 1.0$ for $k = 1, ..., 50$ and $= 0.1$ for $k = 51, ..., 100$ while $R(k) = 0.5$ for all k. Using the simulated data displayed in Figure 4.1, estimates of $x(k)$ can be obtained. These are shown as dashed lines in the top panel of Figure 6.1. Because we have used simulated data, we can in this case compare the estimates with the true x-process, which is shown as a solid line. The lower panel of the same figure shows the estimation error standard deviations for different k's. For the first k's, the variance is relative large. This is because the estimate of $x(k)$ in this case is based on

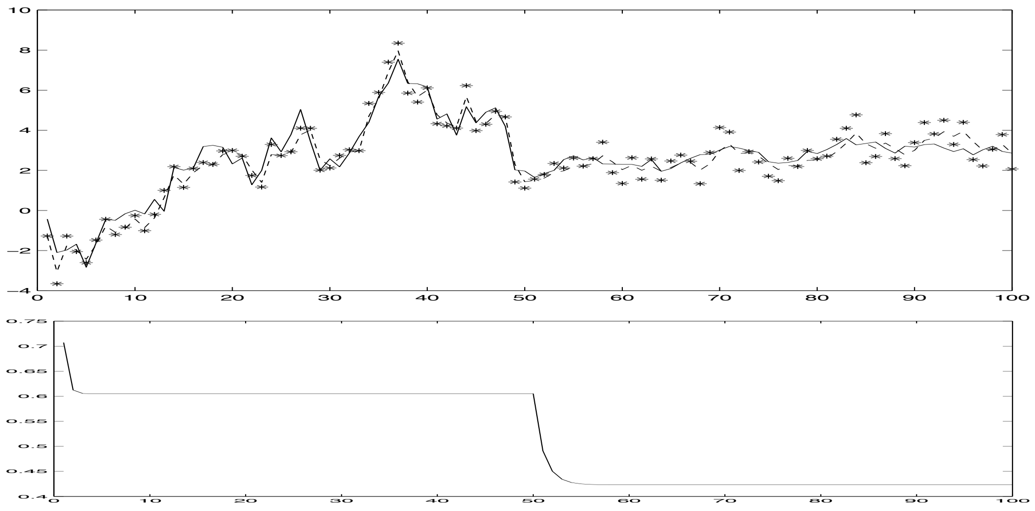

Figure 6.1: In the upper panel, estimates of the random walk process obtained from the Kalman filter based on simulated observations (shown as stars) are displayed. The Kalman filter estimates are shown as a dashed line, while the true system process is shown as a solid line. The lower panel shows the estimation error standard deviations $\sqrt{P(k|k)}$. The simulated data in Figure 4.1 was used.

only a few observations. After 3-4 time steps, the variance seems to stabilize. This is a typical behavior, and will be discussed further in Chapter 8. The large decrease around $k = 50$ is due to that $Q(k)$ decreases from 1 to 0.1 at this point. □

Example 1 (A moving body, cont.)
Assuming only random disturbances acting on the body in Figure 1.1, we get the following discrete representation

$$\begin{pmatrix} x_1(k) \\ x_2(k) \end{pmatrix} = \begin{pmatrix} 1 & T \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x_1(k-1) \\ x_2(k-1) \end{pmatrix} + \begin{pmatrix} w_1(k-1) \\ w_2(k-1) \end{pmatrix} \tag{6.24a}$$

where T is the sampling time. The system noise may represent forces due to wind gusts and turbulence and is assumed to be white and satisfy the other criteria in the state space model (4.6).[1] In the special case $Q_{11}(k) = 0, Q_{22}(k) = 0$, $w_1(k) = 0, w_2(k) = 0$ and so $x_1(k) = a + kTb$, where $a = x_1(0)$ and $b = x_2(0)$ are the initial position and speed at the start. The presence of non-zero random errors means fluctuations and deviations from this deterministic pattern. Such a model, but with movements in

[1] For the body we are actually talking about continuous noise processes that are *discretized*. The details in this case are much more involved than discretization of deterministic inputs and are beyond the scope of these notes. Practical approaches to noise discretization are presented IN 358. A thorough theoretical understanding requires stochastic calculus.

two dimensions simultaneously (so that there are, say x_3 and x_4-processes as well) is actually used in sea navigation. The random terms would then be due to, for example, fluctuations in wind or current. If position is measured, we have

$$z(k) = \begin{pmatrix} 1 & 0 \end{pmatrix} \begin{pmatrix} x_1(k) \\ x_2(k) \end{pmatrix} + v(k). \tag{6.24b}$$

where $v(k)$ (with variance $R(k)$) is the error in measuring $x_1(k)$. If only $x_2(k)$ is measured, the measurement matrix becomes instead $\mathbf{H} = (0 \;\; 1)$ and the measurement error is now, of course, the error in measuring $x_2(k)$.

Figure 6.2 shows simulations of the model with Kalman filter estimates. The simulated observations are indicated by asterisks, while the states and their estimates for convenience are displayed by solid and dotted lines, respectively. (Of course, both the states and their estimates are only defined at each integer k.)

Both initial estimates are zero since in this case $\hat{\mathbf{x}}(0|0) = E\mathbf{x}(0) = \mathbf{0}$. The x_1 estimate follows the real state quite well, while the estimate for x_2 seems to lag behind the true process to some extent. Such lags between estimates and actual state vector components occur frequently. There are two reasons for this. Firstly, there are typically fewer observations than state vector components at each time point. The filter then obtains its reconstructions by combining observations over several time points. Secondly, the innovations in the updating step have to be treated conservatively and be *weighted* properly by the matrix $\mathbf{K}(k)$ because of the measurement noise.

The results in Figure 6.2 represent one realization of the $\mathbf{x}$ and z processes, which again lead to one realization of the filter estimates. The necessary *Matlab* code for this example is available in the Appendix A and we recommend strongly that you play with this example and get a feeling for how variable the realizations may be from case to case. Also alter $\mathbf{Q}$ and R and try to get an intuitive understanding for how the patterns change.

Figure 6.3 shows the error standard deviations for the model in (6.24a) and (6.24b). Note the initial transient phases in the standard deviations before stationary levels are reached. For completeness, the lower panel of Figure 6.3 shows the Kalman filter gains. As one would expect, since the gains are functions of the standard deviations, these elements also display a transient phase before reaching a steady state.

It was noted that the error covariance matrices, and therefore the standard deviations, do not depend on the $\mathbf{z}$ process. Therefore the covariance recursion (6.13) can be carried out and the filter assessed before any actual or simulated measurements are processed. This simplifies the design of a Kalman filter in, for example, an engineering context. There, one typically has to select a certain measurement strategy from many possible ones to meet specified performance and cost criteria. This can be achieved through the covariance recursion trying out various combinations of measurements

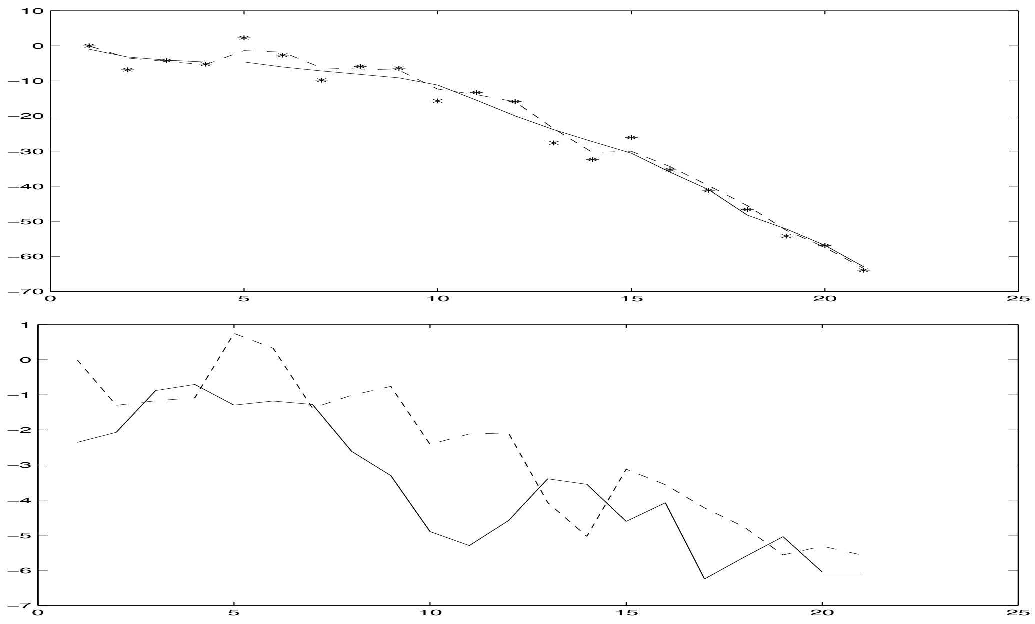

Figure 6.2: One realization of the solution of the model in (6.24a) and (6.24b) with Kalman filter estimates. The true x_1 (upper panel) and x_2 (lower panel) are the solid lines, while the filter estimates are dashed. The measurements are indicated by asterisks. Here $Q_{11} = 1, Q_{22} = 2, Q_{12} = Q_{21} = 0$, and $R = 10$.

(i.e. different $\mathbf{H}$) each with potentially different precision (i.e. different $\mathbf{R}(k)$). As an exercise you may want to run the *Matlab* code in Appendix A to try to achieve stationary standard deviations for the errors in both x_1 and x_2 below 1.

Before leaving this example, you may wonder what the Kalman filter has achieved. How much have we gained compared to a procedure based solely on estimation without any measurement updating? It turns out that this corresponds to running the covariance recursion with $\mathbf{K}(k) = \mathbf{0}$ (see Section 9.1). The standard deviations for the errors in this case are plotted in Figure 6.4. (Note that now $\mathbf{P}(k|k) = \mathbf{P}(k|k-1)$). Convince yourself that what we are now looking at are the errors for an estimator $\hat{\mathbf{x}}(k) = \Phi(k)\hat{\mathbf{x}}(k-1) = \mathbf{0}$ since in this particular case $E[\mathbf{x}(0)] = \mathbf{0}$.) Compared to Figure 6.3, the errors are much larger and they do not reach stationary values but continue to grow. Clearly, the Kalman filter achieves quite a lot. □

Note that this model is sometimes also used in other fields such as economics where it is denoted the linear growth model. Suppose, for example, that $x_1(k)$ is the current value of some economic indicator or variable, say the revenue of a firm or a certain branch of firms. Eq. (6.24a) then says that the sales volume at time t_k is likely to grow by an amount $x_2(k)T$ until the next time point t_{k+1}, a random disturbance $w_1(k)$ coming in addition. The growth term itself is in turn subject to random fluctuations.

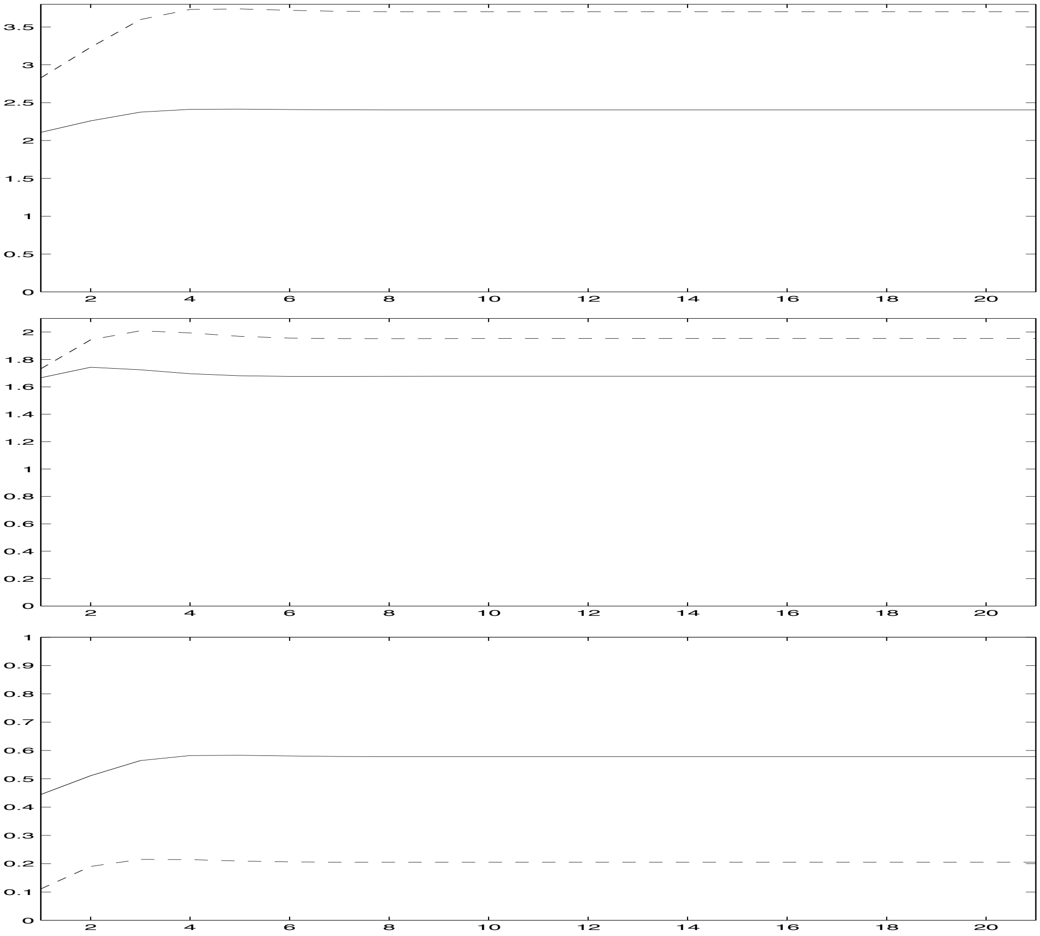

Figure 6.3: Standard deviations for the errors in $\hat{\mathbf{x}}(k|k)$ (solid line) and $\hat{\mathbf{x}}(k|k-1)$ (dashed line), for $x_1(k)$ in the upper panel and for $x_2(k)$ in the middle panel. In the lower panel, the Kalman filter gains K_{11} and K_{12} are displayed. Here $P_{11}(0|0) = 5$, $P_{22}(0|0) = 2$, $Q_{11} = 1, Q_{22} = 2, Q_{12} = Q_{21} = 0$, and $R = 10$.

The formulation makes sense economically since it allows us to express mathematically that growth (or decline) in sales are expected, though not in a quite foreseeable way, as in the real world.

When it comes to the observations, it would in navigation be possible to measure position $x_1(k)$ and/or speed $x_2(k)$. In the economic sphere, probably only the sales volume $x_1(k)$ could be observed. In the latter case, the measurement equation becomes as in (6.24b).

It is worthwhile to note in passing that the model in eqns. (6.24a) and (6.24b) reduces to an ordinary regression line if $Q_{11}(k) = Q_{22}(k) = 0$. As remarked above, we may then write $x_1(k) = a + kTb$, where $a = x_1(0)$ is initial position and $b = x_2(0)$ is the (constant) speed. Hence, $z(k) = a + kTb + v(k)$, and the observations are thus

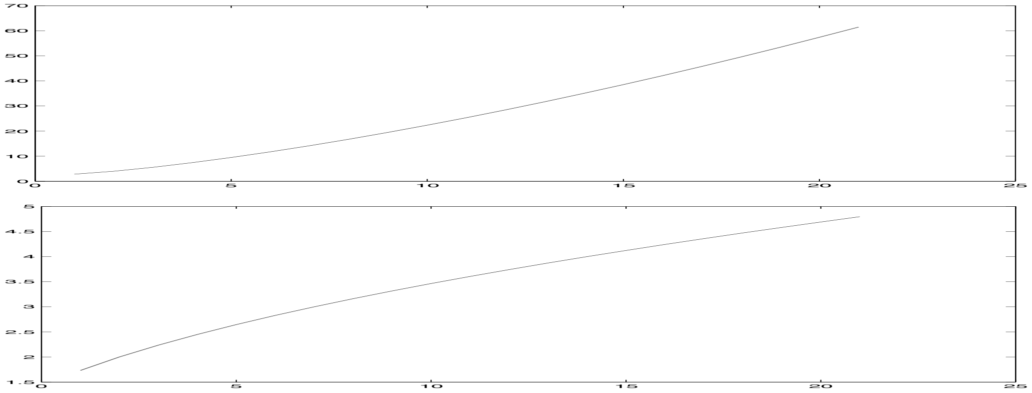

Figure 6.4: Standard deviations for the estimation errors in x_1 and x_2 obtained by setting $\mathbf{K}(k) = \mathbf{0}$.

measurements around a regression line.

6.5 The Kalman filter under non-Gaussian noise

Although the algorithm of Subsection 6.2 was derived under Gaussian assumptions, it remains valid for other types of noise in a weaker sense, with the global optimality lost. Underlying this statement is the fact that the first and second order moments of sums of uncorrelated random vectors are determined by the first and second order moments of the individual terms with distributional assumptions beyond this being immaterial. To make the idea precise, let

$$l_z(\vec{\mathbf{z}}(k)) = \sum_{r=1}^{k} \mathbf{a}(r)\mathbf{z}(r), \tag{6.25}$$

and

$$l_x(\vec{\mathbf{x}}(k)) = \sum_{r=1}^{k} \mathbf{b}(r)\mathbf{x}(r) \tag{6.26}$$

be linear forms in the observations and the system process respectively ($\vec{\mathbf{x}}(k)$ in (6.26) is the vector obtained by collecting $\mathbf{x}(1), ..., \mathbf{x}(k)$ in a way similar to $\vec{\mathbf{z}}(k)$ in (6.3)). Suppose the coefficients $\mathbf{a}(r)$ and $\mathbf{b}(r)$ are matrices with the same number of rows so that the difference $l_z(\vec{\mathbf{z}}(k)) - l_x(\vec{\mathbf{x}}(k))$ is well defined. Then:

Lemma 6.2
The mean and covariance matrix of $l_z(\vec{\mathbf{z}}(k)) - l_x(\vec{\mathbf{x}}(k))$ are determined by the parameters $\{\mathbf{\Phi}(k)\}$, $\{\mathbf{H}(k)\}$, $\{\mathbf{Q}(k)\}$ and $\{\mathbf{R}(k)\}$ of (4.6) and the mean/covariance matrix

of the start vector $\mathbf{x}(0)$ of the $\mathbf{x}$-process, all other distributional assumptions being irrelevant.

This almost immediate result (which is proved at the end of the section) implies that the Kalman filter algorithm of section 6.2 still works with the same error estimates, and with the innovation process having the same properties, except for "uncorrelatedness" replacing "independence". This must be so since the statistics considered there is precisely of the linear form in Lemma 6.2. The only difference when we go to the non-Gaussian case is that the filter is no longer globally the best one, although it is still the best procedure available of *the linear type.* To see this, consider, for simplicity a scalar x-process. Let E represent expectation with respect to some general noise processes whereas E_g is with respect to Gaussian noise *with the same* $\{\mathbf{Q}(k)\}$ *and* $\{\mathbf{R}(k)\}$ *sequences* as in E. Then, with $l(\vec{\mathbf{z}}(k))$ a linear function of $\mathbf{z}(1), .., \mathbf{z}(k)$

$$\begin{aligned} E[(\hat{x}(k|k) - x(k))^2] =& E_g[(\hat{x}(k|k) - x(k))^2] \\ \leq& E_g[(l(\vec{\mathbf{z}}(k)) - x(k))^2] \\ =& E[(l(\vec{\mathbf{z}}(k)) - x(k))^2], \end{aligned}$$

utilizing that $\hat{x}(k|k)$ is optimal under Gaussian models (second line) and that E and E_g, according to lemma 6.2, have the same effect on linear statistics (first and third line). Hence, $\hat{x}(k|k)$ cannot be worse than any other linear method $l_z(\vec{\mathbf{z}}(k))$ as estimator for $x(k)$. This argument can clearly be applied to any element of a *vector* process $\mathbf{x}(k)$.

Proof of Lemma 6.2 Substitute $\mathbf{H}(r)\mathbf{x}(r) + \mathbf{v}(r)$ for $\mathbf{z}(r)$ in the expression for $l_z(\vec{\mathbf{z}}(k))$. Then

$$l_z(\vec{\mathbf{z}}(k)) - l_x(\vec{\mathbf{x}}(k)) = \sum_{r=1}^{k} \mathbf{a}(r)\mathbf{v}(r) + \sum_{r=1}^{k} \mathbf{b}^{(1)}(r)\mathbf{x}(r),$$

where $\mathbf{b}^{(1)}(r)$ are new coefficients, determined by the old $\mathbf{b}(r)$, $\mathbf{a}(r)$ and $\mathbf{H}(r)$. Now, eliminate successively $\mathbf{x}(k)$ (by inserting $\mathbf{\Phi}(k-1)\mathbf{x}(k-1) + \mathbf{w}(k-1)$), then $\mathbf{x}(k-1)$ (using $\mathbf{\Phi}(k-2)\mathbf{x}(k-2)+\mathbf{w}(k-2)$ as replacement) and so forth until all of $\mathbf{x}(k), \mathbf{x}(k-1), ..., \mathbf{x}(1)$ have disappeared, and we are left with

$$l_z(\vec{\mathbf{z}}(k)) - l_x(\vec{\mathbf{x}}(k)) = \sum_{r=1}^{k} \mathbf{a}(r)\mathbf{v}(r) + \sum_{r=1}^{k} \mathbf{b}^{(2)}(r)\mathbf{w}(r-1) + \mathbf{b}^{(2)}(0)\mathbf{x}(0),$$

where $\{\mathbf{b}^{(2)}(r)\}$ is another set of coefficients, related to the former $\mathbf{b}^{(1)}(r)$'s and the $\mathbf{\Phi}(r)$'s. The conclusion of the lemma follows from this. Take, for example, the covariance (which is more complicated than the mean). Since all the random variables on the right are uncorrelated by assumption, the covariance matrix of the sum follows by summing the covariance matrices of the individual terms. Hence,

$$\begin{aligned}\mathrm{VAR}[l_z(\vec{\mathbf{z}}(k)) - l_x(\vec{\mathbf{x}}(k))] = &\sum_{r=1}^{k} \mathbf{a}(r)\mathbf{R}(r)\mathbf{a}(r)^T + \sum_{r=1}^{k} (\mathbf{b}^{(2)}(r))\mathbf{Q}(r-1)(\mathbf{b}^{(2)}(r))^T \\ &+ (\mathbf{b}^{(2)(0)}\mathrm{VAR}[\mathbf{x}(0)](\mathbf{b}^{(2)}(0))^T\end{aligned}$$

showing that only the covariance matrices $\{\mathbf{Q}(r)\}$ and $\{\mathbf{R}(r)\}$ matter for the covariance of $l_z(\vec{\mathbf{z}}(k)) - l_x(\vec{\mathbf{x}}(k))$, as stated. ■

6.6 The Kalman filter for non-zero input variables

As mentioned in the beginning of this chapter, the derivation of the Kalman filter becomes more simple notationally when there are no deterministic input terms available. The mathematics is, however, just as easy as for non-zero input vectors. In this section we will present the recursive algorithms for this case.

The stochastic state model is now defined by (4.5) Note first that since the input terms $\mathbf{u}(k)$ are deterministic and known, they do not influence on the uncertainty involved. The updating equations (6.13) for calculating the covariance matrix $\mathbf{P}(k|k)$ for the estimation error therefore remains unchanged.

Concerning the estimate itself, the modification needed in this case is to add an additional term $\mathbf{\Psi}(k-1)\mathbf{u}(k-1)$ to the prediction estimate $\hat{\mathbf{x}}(k|k-1)$. However, because the input term does not go into the observation model (4.5b), the filtering (or updating) part remains unchanged, giving the following recursive equations:

$$\hat{\mathbf{x}}(k|k-1) = \mathbf{\Phi}(k-1)\hat{\mathbf{x}}(k-1|k-1) + \mathbf{\Psi}(k-1)\mathbf{u}(k-1) \quad \text{(prediction)} \quad (6.27a)$$

$$\hat{\mathbf{x}}(k|k) = \hat{\mathbf{x}}(k|k-1) + \mathbf{K}(k)(\mathbf{z}(k) - \mathbf{H}(k)\hat{\mathbf{x}}(k|k-1)) \quad \text{(updating)} \quad (6.27b)$$

6.7 Problems

Exercise 6.1 (Properties of AR(1) models)
We will in this exercise study a simple state space model with a state variable $x(k)$ and an observation $z(k)$ for $k = 1, \ldots$. Consider the non-observable x-process first. Assume that $x(k) = ax(k-1) + w(k-1)$ for $k = 1, \ldots$ where a is a parameter and $w(k)$ is white noise. It is assumed that $\mathrm{var}[w(k)] = Q$. Also assume that $E[x(0)] = 0$ and $\mathrm{Var}[x(0)] = P_x(1)$.

From Exercise 5.3 we know that $E[x(k)] = 0$ and $P_x(k) = \mathrm{Var}[x(k)]$ is given by $P_x(k) = a^2P_x(k-1) + Q$. Further, $P_x(k) = Q\sum_{j=1}^{k} a^{2(j-1)} + a^{2k}P_x(1)$.

(a) What happens with $P_x(k)$ when $k \to \infty$? Handle the cases $|a| < 1$, $|a| = 1$ and $|a| > 1$ separately.

(b) Show that $\text{cov}[x(k), x(k-j)] = a^j P_x(k-j)$. Decide from this, for a fixed j and $|a| < 1$, the limit $\lim_{k\to\infty} \text{Cov}[x(k), x(k-j)]$,

(c) Specify the correlation $\text{corr}[x(k), x(k-j)]$. Also for the correlation, calculate the limit $\lim_{k\to\infty} \text{Corr}[x(k), x(k-j)]$, for $|a| < 1$.

(d) Assume the process $\{x(k)\}$ has been running for a long time. How would you from the calculations above characterize the statistical properties of the process when $|a| < 1$? How does a influence on the behavior of the process?

(e) Simulate the process in *Matlab* for different values of a, f.ex. $a = -2, -0.8, 0, 0.5, 0.8, 1, 2$. Use $Q = 1$.

Exercise 6.2 (Analytical calculations of Kalman filter)
We will consider the same process as in Exercise 6.1. In practice, $x(k)$ is only observed indirectly through the process $\{z(k)\}$. Assume $z(k) = hx(k) + v(k)$ for $k = 1, 2, \ldots$ where $v(k)$ is white noise with $\text{Var}[v(k)] = R$. The two noise processes $\{w(k)\}$ and $\{v(k)\}$ are assumed to be independent of each other. This implies that $\{v(k)\}$ and $\{x(k)\}$ are independent if $\{v(k)\}$ is independent of $x(1)$. We will make this assumption.

(a) Calculate $E[z(k)]$ and $\text{Var}[z(k)]$? What happens with $\text{Var}[z(k)]$ when $k \to \infty$? Discuss this for different values of a.

A quantity of interest when analyzing series of data is the so-called **signal to noise ratio**. In our context it is defined as

$$SR = \frac{\text{var}[hx(k)]}{\text{var}[v(k)]},$$

that is the ratio of the variances between the two terms which is contained in $z(k)$. The nominator is the variance of the term “explained” by the underlying system model (the signal) while the second term is the variance of the “unexplained” or noise term.

(b) What is the signal to noise ratio in the particular model under study? You may consider the case $k \to \infty$ if you find this relevant.

Let $\hat{x}(k|k-1)$ be the optimal estimate of $x(k)$ based on the observations $z(1), \ldots, z(k-1)$ and let $\hat{x}(k-1|k-1)$ be the optimal estimate of $x(k-1)$ based on the same observations. Assume all processes are Gaussian.

(c) Consider the following question: Which variables in the model is it natural to assume are Gaussian?

(d) Show that $\hat{x}(k|k-1) = a\hat{x}(k-1|k-1)$.

(e) Find the innovation process. What are the most important statistics properties of this process? Discuss and give mathematical expressions where you find it relevant.

(f) Apply the Kalman filter to calculate $\hat{x}(k|k)$ based on $\hat{x}(k|k-1)$ and the innovation. Find a recursive formula for the Kalman gain.

We will assume there is no system noise in the following, that is $w(k) = 0$.

(g) Try to solve the recursion for the Kalman gain by finding an expression for it.

(h) Also find an expression for the variance to the estimation error $\hat{x}(k|k) - x(k)$, that is $P(k|k) = E[(\hat{x}(k|k) - x(k))^2]$

(i) Discuss how the expressions in (g) and (h) depend on a.

(j) Try now to vary Q and R and inspect how the stationary value for the Kalman gain changes. Do the results fit with your intuition?

Exercise 6.3 (Applications of the Kalman filter)
We will consider further the system discussed in Exercises 6.1 and 6.2. In Exercise 6.2 (g) we ended up with the recursive formulae for the Kalman filter.

(a) Try to make a *block-diagram* for the process and the filter (Hint: The `simulink` system in *Matlab* can be used for drawing block-diagrams.)

(b) Explain which part that is the "real world" and which part that takes place in the computer. Why is it that the Kalman filter can be perceived as a process model which runs in parallel with the system? How is it "corrected" or updated in the filter?

(c) Simulate the system without the Kalman filter in *Matlab* . Let N (the number of observations) be 100. Use $a = 0.9$, $h = 1$, $Q = 1$ and $P_x(1) = 1$. Try both $R = 0.1$ and $R = 1$. Plot the x-process as a solid line and the z-process as point around the state.

(d) Apply the Kalman filter to reconstruct the x process from the observations. Try for both values of R. Plot the true and estimated x-process in the same figure. You may also add the observations.

(*e*) Plot the Kalman gain and the variance of the estimation error $P(k|k)$. Justify that $P(k|k)$ is correct by generating many realizations of the estimation error and estimate its variance and mean at each time point for some k's.

(*f*) Now put $P_x(1) = 0$ and $P_x(1) = 100$. Again calculate the Kalman gain and corresponding $P(k|k)$'s. Compare and interpret what you see.

(*g*) It looks like the Kalman gain and the variance of the estimation error approaches some stationary values for increasing n. Argue that this is reasonable. Find the stationary values analytically and compare with the numerical values.

Chapter 7

Examples from statistics

The purpose of this chapter is to demonstrate how linear state space modeling unify linear statistical techniques such as linear regression, analysis of variance and time series (ARMA) models. It is good training to work out how these models can be put in state space form. It is also practically useful in that the superior data processing capability of the Kalman filter becomes available. Missing observations is a case in point. Many papers in statistics have in the past been devoted to this issue for particular models, but the simplest and most general way is to let the Kalman filter take care of it. This requires the model to be written in state space form. We shall consider three main types of situations. Linear regression with one explanatory variable is treated in Section 7.1, while multiple regression is the topic of Section 7.2. The chapter is closed in Section 7.3 with a discussion of time series models.

7.1 Regression with one explanatory variable

Basic formulation The traditional linear regression model relates response $z(k)$ and explanatory variable $x(k)$ through

$$z(k) = \alpha + \beta x(k) + v(k), \tag{7.1}$$

where α and β are coefficients and $v(k)$ independent errors with mean 0. (7.1) can be written in state space form. Formally, take $\alpha(k) = \alpha$ and $\beta(k) = \beta$ and note the trivial relationship

$$\begin{pmatrix}\alpha(k)\\ \beta(k)\end{pmatrix} = \begin{pmatrix}1 & 0\\ 0 & 1\end{pmatrix}\begin{pmatrix}\alpha(k-1)\\ \beta(k-1)\end{pmatrix} + \begin{pmatrix}0\\ 0\end{pmatrix} \tag{7.2a}$$

which can be used as the system equation. The measurement equation is (7.1), or when rewritten in state space form

$$z(k) = \begin{pmatrix}1 & x(k)\end{pmatrix}\begin{pmatrix}\alpha(k)\\ \beta(k)\end{pmatrix} + v(k). \tag{7.2b}$$

Note that $\{x(k)\}$ in this case are *known* coefficients without errors. Regression data can then be fit by running the Kalman filter. The estimates $\hat{\alpha}(N|N)$ and $\hat{\beta}(N|N)$ at the end of the series coincides with the ordinary least squares estimates. This must be so since Kalman filtering is optimal in least squares sense and since $\alpha = \alpha(N)$ and $\beta = \beta(N)$. Although the exercise may seem pointless since there is already a satisfactory numerical procedure available, the state space based approach is much easier to generalize.

Bayesian regression Recall that (7.2a) amounts to a constant sequence of $\alpha(k)$, $\beta(k)$, the values being defined by the initial values. The Kalman filter is fed information of the form

$$\begin{aligned} E[\alpha(0)] &= \alpha_0, & \operatorname{var}[\alpha(0)] &= \sigma_\alpha^2 \\ E[\beta(0)] &= \beta_0, & \operatorname{var}[\beta(0)] &= \sigma_\beta^2 \end{aligned}$$

defining what is known about the regression coefficients in advance[1]. The means α_0, β_0 and the variances σ_α^2, σ_β^2 are under the control of the user. The traditional least squares solution is obtained by letting $\sigma_\alpha \to \infty$ and $\sigma_\beta \to \infty$. For finite σ_α and σ_β we have Bayesian analysis where what is known about the regression coefficients in advance is combined with information from the data.

Autocorrelated errors Another possibility is to work with models where the error terms $v(k)$ in (7.1) are serially correlated (colored noise), which is often of importance in economics and for industrial data. Autocorrelated errors will be discussed more generally in section 7.3, but let us here consider a simple example. Assume $v(k) = av(k-1) + \varepsilon(k-1)$, which is the *autoregressive* model of order 1. The regression model (7.1) with such a specification of the errors can be represented in state space form by adding a line to (7.2a), i.e.

$$\begin{pmatrix} \alpha(k) \\ \beta(k) \\ v(k) \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & a \end{pmatrix} \begin{pmatrix} \alpha(k-1) \\ \beta(k-1) \\ v(k-1) \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ \varepsilon(k-1) \end{pmatrix}$$

and replacing (7.2b) by

$$z(k) = \begin{pmatrix} 1 & x(k) & 1 \end{pmatrix} \begin{pmatrix} \alpha(k) \\ \beta(k) \\ v(k) \end{pmatrix}.$$

The state space approach also has a potential for allowing the regression coefficients to fluctuate in time. This is very simply accommodated by replacing zeros on the

[1] A possible initial correlation $\operatorname{corr}[\alpha(0), \beta(0)]$ is neglected in this case.

right in the preceding system equation by random terms. The effect when estimating $\alpha(k)$ and $\beta(k)$ from a record of observations is that less credence is assigned to old data than more recent ones. The state space approach thus yields a prescription for downgrading information. This aspect is much discussed in forecasting, and several methods proposed there are in reality special cases of Kalman filtering.

7.2 Multiple linear regression

The construction in the preceding subsection extends immediately to p explanatory variables $x_1(k), ..., x_p(k)$. Replace (7.1) by

$$z(k) = \alpha + \beta_1 x_1(k) + \ldots + \beta_p x_p(k) + v(k), \tag{7.3}$$

which can be put into state space form in the manner described in the preceding subsection. To see this replace (7.2a) by

$$\begin{pmatrix} \alpha(k) \\ \beta_1(k) \\ \vdots \\ \beta_p(k) \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{pmatrix} \begin{pmatrix} \alpha(k-1) \\ \beta_1(k-1) \\ \vdots \\ \beta_p(k-1) \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}. \tag{7.4}$$

All the remarks in the preceding subsection applies to this multiple regression model as well. We shall here briefly discuss another aspect. It is well known that multiple linear regression contains a number of special cases, of high interest, for example analysis of variance and analysis of covariance, which comes out when some (or all) of the explanatory variables $\{x_j(k)\}$ are binary. Consider an example where each unit on which the response $z(k)$ has been observed, belongs to one of g groups. There is an additional explanatory variable, called $x_{g+1}(k)$ for reasons given shortly. A model often used in such a situation[2] is

$$z(k) = \beta_i + x_{g+1}(k)\beta_{g+1} + v(k), \qquad \text{if } k \text{ belongs to group } i. \tag{7.5}$$

There are separate parameters β_i characterizing the groups and an effect $\beta_{g+1} x_{g+1}(k)$ from the additional variable. By defining for each i

$$x_i(k) = \begin{cases} 1 & \text{if unit } k \text{ belongs to group } i, \\ 0 & \text{otherwise,} \end{cases}$$

the model can be written

$$z(k) = \beta_1 x_1(k) + \ldots + \beta_g x_g(k) + \beta_{g+1} x_{g+1}(k) + v(k),$$

[2] leading to an analysis often called *analysis of covariance*.

clearly a special case of (7.3). This type of construction can be extended to several grouping variables so that any analysis of variance model can be handled by (7.3). Note that this, in turn, yields all the benefits detailed in the preceding subsection. In particular, so–called model II cases (where the β-coefficients are stochastic), can be handled, and this is advantageous since they may, in the traditional setting, be somewhat cumbersome computationally.

7.3 Time series models

Much statistical modeling is concerned with data containing a time element. The examples to follow are special cases of a huge body of techniques known in statistics as time series models. A common view, especially in economics, is to regard the observed data, say $z(k)$, as a superposition of three different effects. Thus assume

$$z(k) = T(k) + S(k) + v(k), \tag{7.6}$$

where $T(k)$ is a so–called *trend*, accounting for, in some sense, persistent growth (or decline), $S(k)$ represents *seasonality*, i.e variations due to annual, monthly, weekly or even daily cycles, whereas $v(k)$ stands for short-term fluctuations caused by other factors. Although there is in a mathematical sense no clear-cut distinction between the three effects, they represent nevertheless a useful conceptualization for modeling. We shall now present several different formulations, and demonstrate how they can be put into a state space form.

Growth Suppose first that there are no seasonal effects so that (7.6) reads

$$z(k) = T(k) + v(k), \tag{7.7}$$

One way to represent the trend $T(k)$ is to assume that

$$T(k) =T(k-1) + \beta(k-1) + w_1(k-1) \tag{7.8a}$$
$$\beta(k) =\beta(k-1) + w_2(k-1), \tag{7.8b}$$

which coincides (in different notation) with the system process of the linear growth model in Section 6.4 (see the discussion after Example 1). If the short-term fluctuations are modeled as an independent (white noise) process, we have the same linear growth model as in Section 4.1, and the discussion there is valid in the present sequel too. Suppose, however, that the v-process in (7.6) fails to satisfy the independence assumption. It may in practice easily happen that such serial correlations represent important effects. For example, short–term fluctuations in economics are rarely independent from one point in time to the next one. One possible formulation is through an autoregressive process of order 1, usually denoted $AR(1)$). Then $v(k)$ is given by the recursion

$$v(k) = av(k-1) + \varepsilon(k-1). \tag{7.9}$$

Here the coefficient a satisfies $|a| < 1$ and $\varepsilon(k)$ is a white noise process as before. (7.9) is in engineering often called a *Markov process*. It can be proved that the process, after starting far back, reaches a stationary condition where the distribution is the same for all k (see Exercise 5.3). This is usually a reasonable modeling assumption to make. The mean is then 0, and it can under these circumstance also be established that

$$\text{corr}[v(k), v(k+\tau)] = a^\tau, \qquad \tau > 0$$

independent of k. The model can capture serial correlations by adjusting the coefficient[3] a.

Suppose we want to work with a model which is a combination of (7.7), (7.8) and (7.9). Such a model can be put into a state space form as follows. Take as the system equation

$$\begin{pmatrix} T(k) \\ \beta(k) \\ v(k) \end{pmatrix} = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & a \end{pmatrix} \begin{pmatrix} T(k-1) \\ \beta(k-1) \\ v(k-1) \end{pmatrix} + \begin{pmatrix} w_1(k-1) \\ w_2(k-1) \\ \varepsilon(k-1) \end{pmatrix},$$

and let

$$z(k) = \begin{pmatrix} 1 & 0 & 1 \end{pmatrix} \begin{pmatrix} T(k) \\ \beta(k) \\ v(k) \end{pmatrix}$$

be the measurement equation. Note that $v(k)$ is made part of the system process by this construction. Also note that there is no error term in the measurement equation. This is perfectly legal and corresponds to $R(k) = 0$ in the general model of Section 4.2. Note that even though $R(k)$ is not positive definite in this case, the inverse of $S(k)$ will exist. The variance matrix of the three error terms $w_1(k)$, $w_2(k)$ and $\varepsilon(k)$ could be

$$\begin{pmatrix} Q_{11} & Q_{12} & 0 \\ Q_{21} & Q_{22} & 0 \\ 0 & 0 & Q_{33} \end{pmatrix},$$

ignoring a possible dependence on k. Note the zeros which express a likely independence between $v(k)$ and the error terms $w_1(k)$ and $w_2(k)$.

[3] If the condition that $|a| < 1$ is violated, the process either, if $|a| = 1$, wander over the whole real line, coming occasionally infinitely far from the mean value 0, or, if $|a| > 1$ become explosive and turns to infinity.

Seasonality Consider seasonal effects next. As an extremely simple example, suppose there are only two seasons, say summer and winter or day and night. It seems reasonable to describe the effect of seasonality as opposite numbers for each season, i.e.

$$S(k) = \begin{cases} \delta, & \text{if summer,} \\ -\delta, & \text{if winter.} \end{cases}$$

This can also be written $S(k) = -S(k-1)$, or, if slight perturbations in this deterministic pattern is allowed

$$S(k) = -S(k-1) + w_3(k-1), \tag{7.10}$$

where $w_3(k)$ is another error process. The model obtained when this seasonal effect is added to the former model (7.7–7.9), has a state space representation

$$\begin{pmatrix} T(k) \\ \beta(k) \\ S(k) \\ v(k) \end{pmatrix} = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & a \end{pmatrix} \begin{pmatrix} T(k-1) \\ \beta(k-1) \\ S(k-1) \\ v(k-1) \end{pmatrix} + \begin{pmatrix} w_1(k-1) \\ w_2(k-1) \\ w_3(k-1) \\ \varepsilon(k-1) \end{pmatrix},$$

$$z(k) = \begin{pmatrix} 1 & 0 & 1 & 1 \end{pmatrix} \begin{pmatrix} T(k) \\ \beta(k) \\ S(k) \\ v(k) \end{pmatrix}.$$

In practice, there are often more than two seasons (for example four in quarterly data or twelve in monthly ones). These situations can also be expressed in state space form, although with more involved representations.

Autoregression Take $T(k) = \mu$ (μ constant), remove the seasonality and concentrate on the modeling of short–term fluctuations. (7.6) now reads

$$z(k) = \mu + v(k) \tag{7.11a}$$

There is an enormous statistical literature on models for $v(k)$. We have already met the $AR(1)$–model in (7.9). An important extension is

$$v(k) = a_1 v(k-1) + a_2 v(k-2) + \ldots + a_p v(k-p) + \varepsilon(k-1) \tag{7.11b}$$

which exhibits stationary behavior in the sense described earlier when the coefficients $a_1, \ldots, a_p$ satisfy a mathematical condition that has been worked out and is available in standard textbooks on the subject (an alternative is to invoke Theorem 4.1). (7.11b) is known as an $AR(p)$ model. A state space representation of (7.11a), (7.11b) can easily be constructed, for example for $p = 4$:

$$\begin{pmatrix} T(k) \\ v(k) \\ v(k-1) \\ v(k-2) \\ v(k-3) \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & a_1 & a_2 & a_3 & a_4 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} T(k-1) \\ v(k-1) \\ v(k-2) \\ v(k-3) \\ v(k-4) \end{pmatrix} + \begin{pmatrix} 0 \\ \varepsilon(k-1) \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

$$z(k) = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} T(k) \\ v(k) \\ v(k-1) \\ v(k-2) \\ v(k-3) \end{pmatrix}.$$

The system process has five components, three of them being earlier numbers of the v–process. That is perfectly legal and so is the many zeros in the five–dimensional error process (a zero means that the corresponding variance is 0). This exercise is useful in practice, since the Kalman filter is by far the most convenient way of filling out gaps in the series. Technically this is accomplished along the line discussed in Section 4.3.

Moving average State space representations are probably even more useful for the other, main linear model for stationary time series, the so–called moving average (MA) model. Replace (7.11b) with

$$v(k) = \varepsilon(k) + b_1\varepsilon(k-1) + \ldots + b_q\varepsilon(k-q), \tag{7.12}$$

where $\varepsilon(k)$, as usual, is a white noise process. (7.12) is, for short, designated MA(q). The following state space representation is available for (7.12), say when $q = 3$:

$$\begin{pmatrix} T(k) \\ \varepsilon(k) \\ \varepsilon(k-1) \\ \varepsilon(k-2) \\ \varepsilon(k-3) \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} T(k) \\ \varepsilon(k-1) \\ \varepsilon(k-2) \\ \varepsilon(k-3) \\ \varepsilon(k-4) \end{pmatrix} + \begin{pmatrix} 0 \\ \varepsilon(k) \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

$$z(k) = \begin{pmatrix} 1 & 1 & b_1 & b_2 & b_3 \end{pmatrix} \begin{pmatrix} T(k) \\ \varepsilon(k) \\ \varepsilon(k-1) \\ \varepsilon(k-2) \\ \varepsilon(k-3) \end{pmatrix}.$$

The point of the exercise is again that the Kalman filter is offered to perform computations. MA–models are harder to handle technically by other methods than AR-models were. Combination models containing both AR and MA terms (so–called autoregressive moving average (ARMA) models) are often useful, and extensively discussed in statistical literature. State space formulations are again useful, although omitted here.

7.4 Problems

Exercise 7.1 (Linear regression with autocorrelated noise)
Consider the linear regression model with autocorrelated errors in Section 7.1 with $x(k) = \sqrt{k}$. Our aim is to estimate α and β.

(a) Write a general routine for the Kalman filter for this model.

(b) Why is there no use in performing smoothing on such a model?

(c) Simulate a set of realizations of length $N = 20$ from the model with $\alpha = 0.0, \beta = 1.0, Q = \text{var}[\varepsilon(k)] = 1$ and $a = -1.0, -0.75, 0.5, 0.0, 0.25, 0.5, 0.75, 1.0$. For each set of realizations, estimate α and β.

(d) Make a plot of $|\hat{\alpha} - \alpha|$ against a. Do you see any pattern?

(e) Now make a plot against a of the standard deviations for the errors involved in estmating α. Comment on the result. In particular, compare the behavior for $a < 0$ with $a = 0$. Try to understand this.

(f) Repeat (d) and (e) but now consider $\hat{\beta}$.

Exercise 7.2 (AR modeling)
Fitting autoregressive models to data serier is a large field and *Matlab* is well equipped here, such as the command `ar` (try `help ar` and `help theta` to get some information on how to use this routine). We will consider the difference equation

$$z(k) + cz(k-1) + dz(k-2) = w(k-1), \quad k = 2, 3, \ldots$$

where $c = 0.8$, $d = 0.15$ and $w(k-1)$ for a given k is a Gaussian distributed random variable $\mathcal{N}(0, 0.2)$. Let $z(0)$ and $z(1)$ be Gaussian $\mathcal{N}(5, 1)$.

(a) Generate some realizations of $z(k)$ for $k = 0, \ldots, 100$ and plot the results.

(b) Write the model in state space form.

(c) You should have ended up with a model with no measurement noise. In such cases we are not guaranteed that $\mathbf{S}(k)$ can be inverted (see Section 6.2). Make sure that there will be no problems in this case.

(d) Show that the system is asymptotically stable. How do you think this property influence on the behavior of the sequence $\{z(k)\}$.

(e) Show that the system is observable and stochastic reachable.

Chapter 8

Additional aspects of the Kalman filter

In this chapter we will discuss various aspects concerning the Kalman filter. In Section 8.1, we will give a brief discussion of the important topic of stability of the Kalman filter algorithm. For a time – invariant system with constant noise covariance matrices $(\mathbf{Q}, \mathbf{R})$, stability means that the error covariance matrix $\mathbf{P}(k|k)$ approaches a unique matrix $\overline{\mathbf{P}}$,independent of the initial condition, as k increases (note that this is a *different* matrix than the one discussed in Section 4.2). This is important in its own right and is also necessary for numerical errors not to accumulate during the recursions.

Following on from this is a brief treatment of different but algebraically equivalent covariance recursions (Section 8.2). Several versions exist and some of them give additional insight into how the Kalman filter works. We also mention recursions that effectively compute the square root of the covariance matrix, thus limiting the numerical range of the matrix elements involved. Such recursions generally have better numerical properties and may be important in real time applications.

So far, we have assumed that our model (comprising the state space equations and statistical assumptions) are sufficiently accurate to neglect *model* errors. However, in any real application the model forming the basis for the Kalman filter will differ from the "reality" giving rise to the observations. This may lead to the problem of filter divergence, where the diagonal terms of the filter error covariance matrix give a completely wrong picture of the performance of the filter. Typically, the diagonal terms would indicate that the filter works properly, while the estimates in reality have much larger errors. An assessment of the real filter performance is central in any application and some useful techniques will be discussed in the final Section 8.3.

8.1 Stability of the Kalman filter algorithm

The stability results are cumbersome to prove and we will only show an example and state some main facts. For more throughout discussion, we refer to [8].

Example 1 (A moving body, cont.)
Continuing with our two–dimensional moving body example, assume now that only the component x_2 is measured (in contrast to our discussion in Section 6.4 where x_1 was measured). Figure 8.1 shows the estimation error standard deviations in this case. Now only the standard deviation for the error in x_2 reaches a stationary value, while the error in x_1 *diverges.* (Verify this by trying more iterations yourself.) Although the filter still does a lot better than pure prediction from initial conditions alone (cf. Figure 6.4), the error divergence is undesirable; in a sense the filtering problem is ill–posed. □

The reason for the problem in this example turns out to be that the system is not *observable* (cf. Section 2.2). It is interesting that the deterministic property of observability plays a role in stochastic filtering but, with hindsight, this may not be too surprising. It can be shown that if a deterministic system is unobservable, a component of the state vector within a certain subspace has no effect on the observations (in our example above, the state x_1 could have been removed from the system equation without changing the distribution of the observations). Running the Kalman filter in the stochastic case, it seems reasonable that the uncertainty associated with the unobservable state component will depend on the initial uncertainty and possibly grow with time. We will see shortly that the deterministic reachability concept also comes into play.

Consider the general state space model in eqns. (4.6a) and (4.6b). We will only state results for time–invariant systems; i.e. $\mathbf{\Phi}, \mathbf{H}, \mathbf{R}$, and $\mathbf{Q}$ are all constant matrices independent of k. Assume that the system is observable in the sense of Definition 2.2; i.e. the matrix

$$\mathbf{W}_o = \begin{pmatrix} \mathbf{H} \\ \mathbf{H\Phi} \\ \vdots \\ \mathbf{H\Phi}^{n-1} \end{pmatrix}$$

has rank n. Further, if we formally apply the reachability criterion with $\mathbf{w}(k-1)$ as the input, we have $\mathbf{W}_c = \begin{pmatrix} \mathbf{I}_n & \mathbf{\Phi} & \cdots & \mathbf{\Phi}^{n-1} \end{pmatrix}$ which clearly has rank n. The system is therefore what is denoted **stochastically reachable**. Intuitively, this means that the noise affects the whole state space. (Note that there exist other formulations of the stochastic state space model that are not necessarily stochastically reachable.)

We then have the following result:

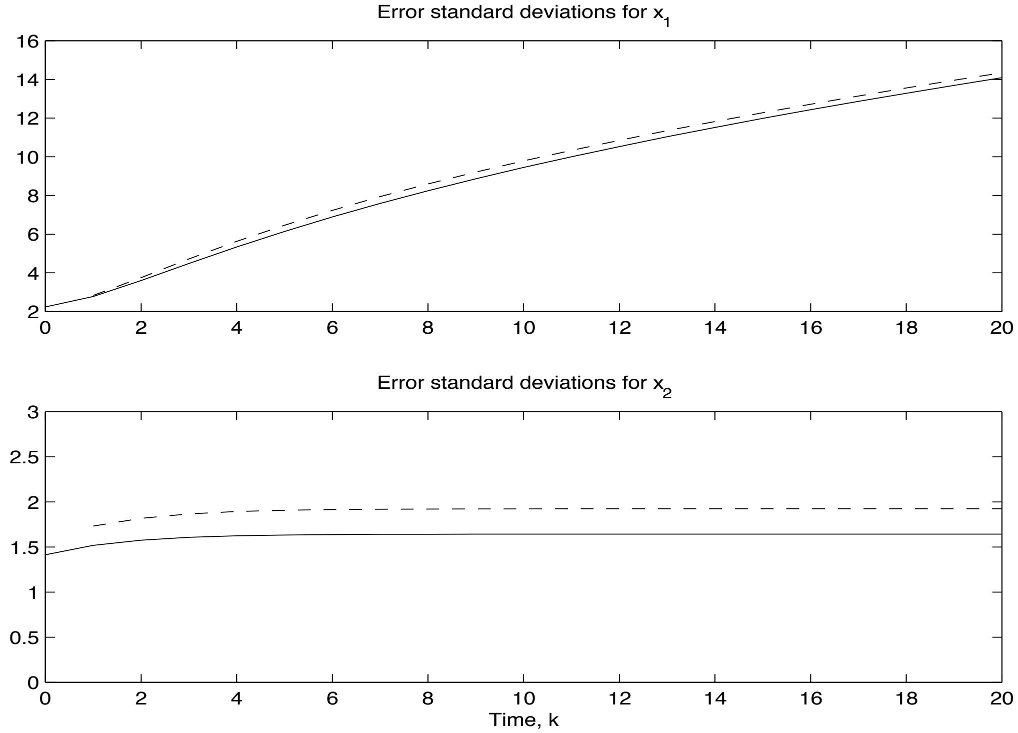

Figure 8.1: Standard deviations for the errors in $\hat{\mathbf{x}}(k|k)$ (solid) and $\hat{\mathbf{x}}(k|k-1)$ (dashed). Here only x_2 is measured. The parameters are as in Figures 6.2 and 6.3.

Theorem 8.1
Assume the system in eqns. (4.6) is time–invariant , observable, and stochastically reachable. Furthermore, $\mathbf{R}$ *and* $\mathbf{Q}$ *are assumed positive definite. Then:*

(a) *For any symmetric and positive definite matrix* $\mathbf{P}(0|0)$, $\mathbf{P}(k|k)$ *converges uniformly to a unique matrix* $\overline{\mathbf{P}}$. *This also implies that* $\mathbf{K}(k)$ *converges to a constant matrix* $\overline{\mathbf{K}}$.

(b) *Define the* **steady state Kalman filter** *as the one using the gain matrix* $\overline{\mathbf{K}}$. *We have* $\hat{\mathbf{x}}(k|k) = \hat{\mathbf{x}}(k|k-1) + \overline{\mathbf{K}}(\mathbf{z}(k) - \mathbf{H}\hat{\mathbf{x}}(k|k-1)) = (\mathbf{\Phi} - \overline{\mathbf{K}}\mathbf{H}\mathbf{\Phi})\hat{\mathbf{x}}(k-1|k-1) + \overline{\mathbf{K}}\mathbf{z}(k)$. *This filter is asymptotically stable, i.e. all eigenvalues of* $(\mathbf{\Phi} - \overline{\mathbf{K}}\mathbf{H}\mathbf{\Phi})$ *lie within the unit circle.*

These results are important in practice since they provide necessary conditions for the error covariance matrices to converge independently of the initial condition $\mathbf{P}(0|0)$. They also guarantee numerical stability of the filter equations. For further details, see [12].

8.2 Equivalent forms of the variance recursion

In Chapter 6, the covariance equations (6.13) were derived. There are several alternative but algebraically equivalent sets of recursive equations. These may sometimes be useful in terms of improved numerical properties or they may be used to enhance the interpretation of the Kalman filter.

8.2.1 The information filter

This section includes some matrix results which are useful to be aware of. To show these results requires a level of proficiency in matrix manipulations that a serious student of Kalman filtering should master. Central is the so-called Matrix Inversion Lemma which states the following:

Lemma 8.1 (Matrix Inversion Lemma)
Let $\mathbf{A}$ $(n \times n)$ *be a nonsingular matrix and let* $\mathbf{B}$ *and* $\mathbf{C}$ *both be* $n \times m$ *matrices. Suppose both* $(\mathbf{A}+\mathbf{B}\mathbf{C}^T)$ *and* $(\mathbf{I}_m + \mathbf{C}^T\mathbf{A}^{-1}\mathbf{B})$ *are nonsingular. Then*

(*a*) $(\mathbf{A}+\mathbf{B}\mathbf{C}^T)^{-1}\mathbf{B} = \mathbf{A}^{-1}\mathbf{B}(\mathbf{I}_m + \mathbf{C}^T\mathbf{A}^{-1}\mathbf{B})^{-1}$.

(*b*) $(\mathbf{A}+\mathbf{B}\mathbf{C}^T)^{-1} = \mathbf{A}^{-1} - \mathbf{A}^{-1}\mathbf{B}(\mathbf{I}_m + \mathbf{C}^T\mathbf{A}^{-1}\mathbf{B})^{-1}\mathbf{C}^T\mathbf{A}^{-1}$.

The proof of the lemma is given in Exercise 8.1. Basically, it allows us to derive the following relationships from the standard variance recursion (see Exercise 8.2 on how to prove these):

$$\mathbf{P}(k|k-1)^{-1} = [\mathbf{I}_n - \mathbf{B}(k-1)]\mathbf{A}(k-1) \tag{8.1a}$$

$$\mathbf{P}(k|k)^{-1} = \mathbf{P}(k|k-1)^{-1} + \mathbf{H}(k)^T\mathbf{R}(k)^{-1}\mathbf{H}(k) \tag{8.1b}$$

$$\mathbf{A}(k) = \mathbf{\Phi}(k)^{-1}\mathbf{P}(k|k)^{-1}\mathbf{\Phi}(k)^{-1} \tag{8.1c}$$

$$\mathbf{B}(k) = \mathbf{A}(k)[\mathbf{A}(k) + \mathbf{Q}(k)^{-1}]^{-1} \tag{8.1d}$$

Eqns. (8.1) are relationships between *inverse* covariance matrices. It allows us to run a recursive algorithm for calculating the inverse of $\mathbf{P}(k|k-1)$. An inverse of a covariance matrix is usually named the **information matrix**. Small noise variances leads to more specific knowledge about the variables, that is more information. The recursions (8.1) are therefore called the **information filter**. The equations above are sufficient for calculating (the inverse of) $\mathbf{P}(k|k)$. In order to calculate the estimates, the Kalman gain is needed. The relationship

$$\mathbf{K}(k) = \mathbf{P}(k|k)\mathbf{H}(k)^T\mathbf{R}(k)^{-1} \tag{8.2}$$

is then available.

A few remarks relating these recursions to the ordinary ones given in (6.13) are in place.

In Section 4.3 we discussed the use of diffuse priors where the diagonal elements of $\mathbf{P}(0|0)$ were put to infinity. When using the ordinary filter, only an approximative solution is possible by substituting infinity with a sufficiently large number. For the information filter, an exact solution is directly available by putting $\mathbf{P}(0|0)^{-1}$ equal to zero.

The information filter requires the inversion of $\mathbf{A}(k)+\mathbf{Q}(k)^{-1}$ (an $n \times n$ matrix), while the ordinary filter require the inversion of $\mathbf{S}(k)$ (an $m \times m$ matrix). A comparison of the computational burden involved for the two filters will therefore depend on n and m.

Because of the need for inversion (and thereby existence) of $\mathbf{\Phi}(k)^{-1}$ and $\mathbf{Q}(k)^{-1}$, the information filter is usually less attractive than the ordinary filter. An important interpretation of the estimation procedure can however be found from (8.1b). Clearly, it shows that $\mathbf{P}(k|k-1)^{-1} \leq \mathbf{P}(k|k)^{-1}$.[1] It can be shown that this is equivalent to $\mathbf{P}(k|k) \leq \mathbf{P}(k|k-1)$ which tells us that $trace(\mathbf{P}(k|k)) \leq trace(\mathbf{P}(k|k-1))$. In other words, the sum of the estimation error variances never increases as a result of a measurement update. This reflects the fact that the uncertainty of an estimate decreases as the amount of data increases.

Intuitively, one sees from (8.1b) that the more accurate the observations are (implying small elements in $\mathbf{R}(k)$ and large elements in $\mathbf{R}(k)^{-1}$), the larger the trace of $\mathbf{P}(k|k)^{-1}$, and the smaller the trace of $\mathbf{P}(k|k)$.

Eq. (8.2) is not very useful from a practical algorithmic point of view. But it tells us that if $\mathbf{P}(k|k)$ has somehow been obtained, the gain matrix can be viewed as a "ratio" between the estimation error variance and the measurement error variance. If, for example, $\mathbf{P}(k|k)$ increases due to increased system noise, then $\mathbf{K}(k)$ will also increase and the measurements are weighted more heavily in the filter measurement update step (for fixed $\mathbf{R}(k)$) so that the residuals become relatively more important.

8.2.2 More versions of the variance recursions

Today, practical Kalman filters are often implemented using covariance recursions based on factorizations of the error covariance matrices. This offers advantages in terms of increased numerical precision without increasing the computation time significantly. Actually, this is a rather well developed field with many available sets of equations. One possibility is to compute a so–called square root $\mathbf{M}(k)$ of the error covariance matrix where $\mathbf{P}(k|k) = \mathbf{M}(k)\mathbf{M}(k)^T$. (Note that square roots of symmetric matrices are not unique.) This effectively doubles the numerical precision because if

[1] Given two matrices $\mathbf{A}$ and $\mathbf{B}$ of the same dimension, $\mathbf{A} \geq \mathbf{B}$ means that $\mathbf{A} - \mathbf{B}$ is positive semidefinite. This in turn implies that $trace(\mathbf{A}) \geq trace(\mathbf{B})$.

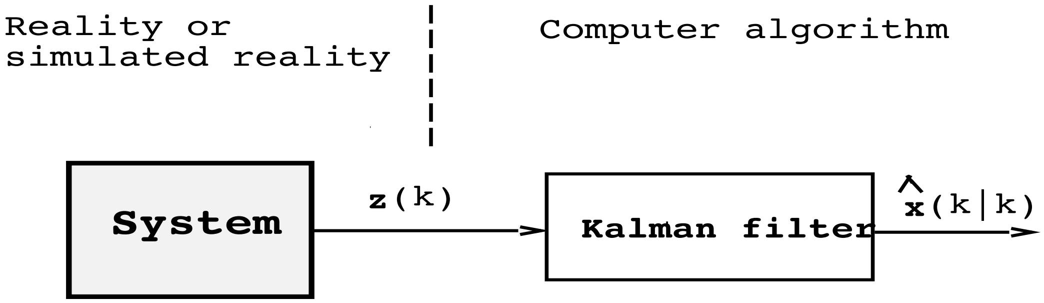

Figure 8.2: The relationship between the Kalman filter and the real system or simulated versions of the real system.

the elements in $\mathbf{P}(k|k)$ range from 2^N to 2^{-N}, those of $\mathbf{M}(k)$ will range from $2^{N/2}$ to $2^{-N/2}$. For more on this see [12]; [6].

8.3 Assessing modeling errors

Our experience is that this topic seems to be conceptually somewhat difficult to grasp. We will therefore take some effort to explain the main ideas. Figure 8.2 shows the relationship between the Kalman filter and the real system whose states the filter estimates. During a design phase, one typically does not process data from the real system but tests the filter on *simulated* versions of the real system.

We imagine now that the system is described by the matrices $\mathbf{\Phi}, \mathbf{H}, \mathbf{R}, \mathbf{Q}, \mu(0)$ and $\mathbf{P}(0|0)$. Under optimal conditions, the filter is based on these same matrices. In this section, the problem is to assess a filter based on $\mathbf{\Phi}^*, \mathbf{H}^*, \mathbf{R}^*, \mathbf{Q}^*$, and $\mathbf{P}^*(0|0)$ processing measurements from the system. You may, of course, ask why anyone would want such a filter. In situations where the correct matrices are known, one then deliberately *chooses* a suboptimal filter rather than the optimal one. This could be the case, for example, if the optimal filter is computationally too demanding in a real time application. However, the most common situation is that the correct matrices are only more or less well known. To design a filter, one has to select a set $\mathbf{\Phi}^*, \mathbf{H}^*, \mathbf{R}^*, \mathbf{Q}^*$, and $\mathbf{P}^*(0|0)$. How then can the performance of this filter be assessed? Well, somebody once said that no amount of simulation trickery can replace missing information, so to really answer this question you simply have to try the filter in reality. But as ardent computer fans, we will still claim that simulations may be useful also in this case because you can pose so–called "what if" questions and get the answers by simulations. For instance, you may think that the diagonal elements of $\mathbf{Q}(k)$ and $\mathbf{R}(k)$ lie between certain boundaries. *If* this is the case and you base your filter on some $\mathbf{Q}^*(k)$ and $\mathbf{R}^*(k)$, the computer will give you the boundaries for the error variances. Doing this, you will hopefully end up entering the real world with a filter that is less likely to fail [2].

In the following, it is convenient to separate between to types of modeling errors.

- The elements in error covariance matrices $\mathbf{P}(0|0), \mathbf{Q}(k)$, and $\mathbf{R}(k)$ may be misspecified but the matrices $\mathbf{\Phi}(k)$ and $\mathbf{H}(k)$ describing the system are correct.
- All matrices may be misspecified and even the dimension of the filter may be different from that of the real system.

8.3.1 Updating the covariance matrix for an arbitrary gain matrix

In some cases one could prefer to use a Kalman gain $\mathbf{K}^*(k)$ different from the optimal choice $\mathbf{K}(k)$. This may be because $\mathbf{K}^*(k)$ is easier to compute or in order to make the procedure more robust. Filters using such gain functions are called *suboptimal* Kalman filters.

Consider the filter equations in eqns.(6.12) and assume that we apply some gain matrix $\mathbf{K}^*(k)$ of our own liking instead of the optimal $\mathbf{K}(k)$. Now using index $*$ to indicate that the estimates are based on $\mathbf{K}^*(k)$, the filter then becomes

$$\hat{\mathbf{x}}^*(k|k-1) = \mathbf{\Phi}(k-1)\hat{\mathbf{x}}^*(k-1|k-1) \qquad \text{(prediction)} \quad (8.3a)$$
$$\hat{\mathbf{x}}^*(k|k) = \hat{\mathbf{x}}^*(k|k-1) + \mathbf{K}^*(k)(\mathbf{z}(k) - \mathbf{H}(k)\hat{\mathbf{x}}^*(k|k-1)) \qquad \text{(updating).} \quad (8.3b)$$

As long as $\mathbf{K}^*(k) \neq \mathbf{K}(k)$, this filter is suboptimal in the sense of Theorem 5.1. But $\mathbf{K}^*(k)$ may still be preferable because we have made it very simple to compute and are therefore willing to accept some destruction in performance compared to the optimal filter. For example, we could set $\mathbf{K}^*(k)$ constant and equal to the stationary gain matrix $\overline{\mathbf{K}}$. Denoting the *real* error covariance matrices involved using estimator $\hat{\mathbf{x}}^*(k|k-1)$ by $\mathbf{P}(k|k-1)$, we now show that the error covariance matrices of the filter in (8.3) are governed by the recursion

$$\mathbf{P}(k|k-1) = \mathbf{\Phi}(k-1)\mathbf{P}(k-1|k-1)\mathbf{\Phi}(k-1)^T + \mathbf{Q}(k-1) \quad (8.4a)$$
$$\begin{aligned}\mathbf{P}(k|k) = &(\mathbf{I}_n - \mathbf{K}^*(k)\mathbf{H}(k))\mathbf{P}(k|k-1)(\mathbf{I}_n - \mathbf{K}^*(k)\mathbf{H}(k))^T \\ &+ \mathbf{K}^*(k)\mathbf{R}(k)\mathbf{K}^*(k)^T. \quad (8.4b)\end{aligned}$$

Eq. (8.4a) follows immediately since the prediction step is the same as for the optimal filter. To see eq. (8.4b), note that

$$\mathbf{x}(k) - \hat{\mathbf{x}}^*(k|k) = (\mathbf{I}_n - \mathbf{K}^*(k)\mathbf{H}(k))(\mathbf{x}(k) - \hat{\mathbf{x}}^*(k|k-1)) - \mathbf{K}^*(k)\mathbf{v}(k).$$

[2] Entering a more philosophical mode, one may take the position that no model is ever really correct. For instance, truly linear systems may never exist. Our model is always an approximation. This is true, so what we have been discussing is the situation where modeling errors are negligible compared to the case where they are significant *as judged by the modeler.*

Taking the covariance matrix on both sides, we obtain the desired relationship.

In the next section we will utilize the recursion eqns. (8.4a) and (8.4b) to assess the performance of a suboptimal filter resulting from *misspecification* of the matrices $\mathbf{P}(0|0), \mathbf{Q}(k)$, and $\mathbf{R}(k)$.

Now consider the case where $K^*(k) = K(k)$. Then we see, somewhat surprisingly, that (6.13d) can also be written as

$$\begin{aligned}\mathbf{P}(k|k) =&(\mathbf{I}_n - \mathbf{K}(k)\mathbf{H}(k))\mathbf{P}(k|k-1)(\mathbf{I}_n - \mathbf{K}(k)\mathbf{H}(k))^T \\ &+ \mathbf{K}(k)\mathbf{R}(k)\mathbf{K}(k)^T. \end{aligned} \tag{8.5}$$

Numerically, an advantage of this equation is that it will ensure a symmetric $\mathbf{P}(k|k)$. As an exercise you may want to start with eq. (8.4b) and derive eq. (6.13a). This will involve some matrix juggling and is good practice.

8.3.2 Misspecified covariance matrices

The suboptimal filter estimates are computed according to

$$\hat{\mathbf{x}}^*(k|k-1) = \mathbf{\Phi}(k-1)\hat{\mathbf{x}}^*(k-1|k-1), \tag{8.6a}$$

$$\hat{\mathbf{x}}^*(k|k) = \hat{\mathbf{x}}^*(k|k-1) + \mathbf{K}^*(k)(\mathbf{z}(k) - \mathbf{H}(k)\hat{\mathbf{x}}^*(k|k-1)). \tag{8.6b}$$

Using the Kalman filter as a black box, we will as output also obtain some covariance matrices (denoted now $\mathbf{P}^*(k|k)$) obtained through the ordinary recursions

$$\mathbf{P}^*(k|k-1) = \mathbf{\Phi}(k-1)\mathbf{P}^*(k-1|k-1)\mathbf{\Phi}(k-1)^T + \mathbf{Q}^*(k-1) \tag{8.7a}$$

$$\mathbf{S}^*(k) = \mathbf{H}(k)\mathbf{P}^*(k|k-1)\mathbf{H}(k)^T + \mathbf{R}^*(k) \tag{8.7b}$$

$$\mathbf{K}^*(k) = \mathbf{P}^*(k|k-1)\mathbf{H}(k)^T\mathbf{S}^*(k)^{-1} \tag{8.7c}$$

$$\mathbf{P}^*(k|k) = (\mathbf{I}_n - \mathbf{K}^*(k)\mathbf{H}(k))\mathbf{P}^*(k|k-1) \tag{8.7d}$$

with initial condition $\mathbf{P}^*(0|0)$, but with $\mathbf{K}(k)$ replaced by $\mathbf{K}^*(k)$. This is the *apparent* error covariance matrices computed by the suboptimal Kalman filter. Referring to Section 8.3.1, it is clear that the *real* error covariance recursion for this filter is given by the formulae in that section:

$$\mathbf{P}(k|k-1) =\mathbf{\Phi}(k-1)\mathbf{P}(k-1|k-1)\mathbf{\Phi}(k-1)^T + \mathbf{Q}(k-1) \tag{8.8a}$$

$$\begin{aligned}\mathbf{P}(k|k) =&(\mathbf{I}_n - \mathbf{K}^*(k)\mathbf{H}(k))\mathbf{P}(k|k-1)(\mathbf{I} - \mathbf{K}^*(k)\mathbf{H}(k))^T \\ &+ \mathbf{K}(k)^*\mathbf{R}(k)\mathbf{K}^*(k)^T \end{aligned} \tag{8.8b}$$

Clearly, $\mathbf{P}^*(k|k), \mathbf{P}(k|k)$, and $\mathbf{P}^0(k|k)$ are all different, where $\mathbf{P}^0(k|k)$ now denotes the error covariance matrix for the *optimal* filter, i.e. the one based on $\mathbf{P}(0|0), \mathbf{Q}(k)$, and $\mathbf{R}(k)$. All we can say is that $P_{ii}(k|k) \geq P^0_{ii}(k|k)$.

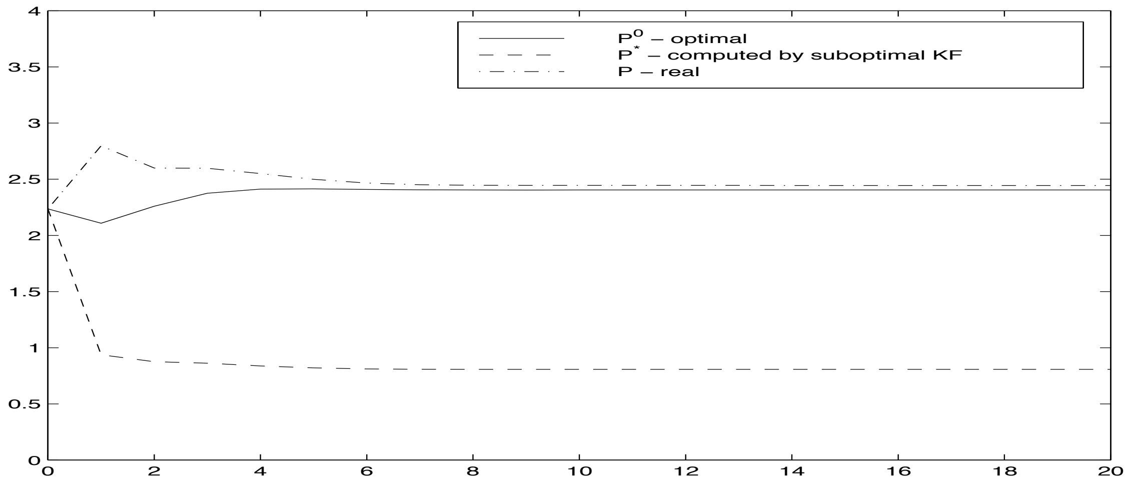

Figure 8.3: Assessing suboptimal filter performance in estimating $x_1(k)$. The *real* error standard deviation is shown together with the standard deviation computed by the suboptimal filter itself. The standard deviation for the *optimal* filter is as in Figure 6.3 and is given as a reference.

Example 1 (A moving body, cont.)
Returning to our two–dimensional example, Figure 8.3 shows three computed error standard deviations.

Here the initial conditions are correct ($\mathbf{P}^*(0|0) = \mathbf{P}(0|0)$), while $R^* = 1$ and $\mathbf{Q}^* = 0.2\mathbf{I}$ compared to $\mathbf{Q} = \mathbf{I}$ and $R = 10$. Thus the filter becomes suboptimal because it is based on too small measurement and system noise variances. As a result, it is overly optimistic regarding its own performance with a stationary standard deviation somewhat below 1. In reality, the true standard deviation is around 2.5 and that is almost as low at it can be comparing with the optimal filter. Again you may want to play with this example using the *Matlab* code given in the Appendix. □

8.3.3 General modeling errors

There exists recursions to compute the real error covariance matrix when all matrices, including $\mathbf{\Phi}(k)$ and $\mathbf{H}(k)$ are misspecified. However, these are rather cumbersome and do not provide much insight. Brute force Monte Carlo simulations are often the best approach. To illustrate this, assume that the filter designer now includes air resistance into the model of the body in Figure 1.1 on which the Kalman filter is based. However, in reality this is negligible (as has been assumed up till now). The filter is based on the model

$$\begin{pmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 0 & -f \end{pmatrix} \begin{pmatrix} x_1(t) \\ x_2(t) \end{pmatrix} + \begin{pmatrix} 0 \\ 1/m \end{pmatrix} u(t)$$

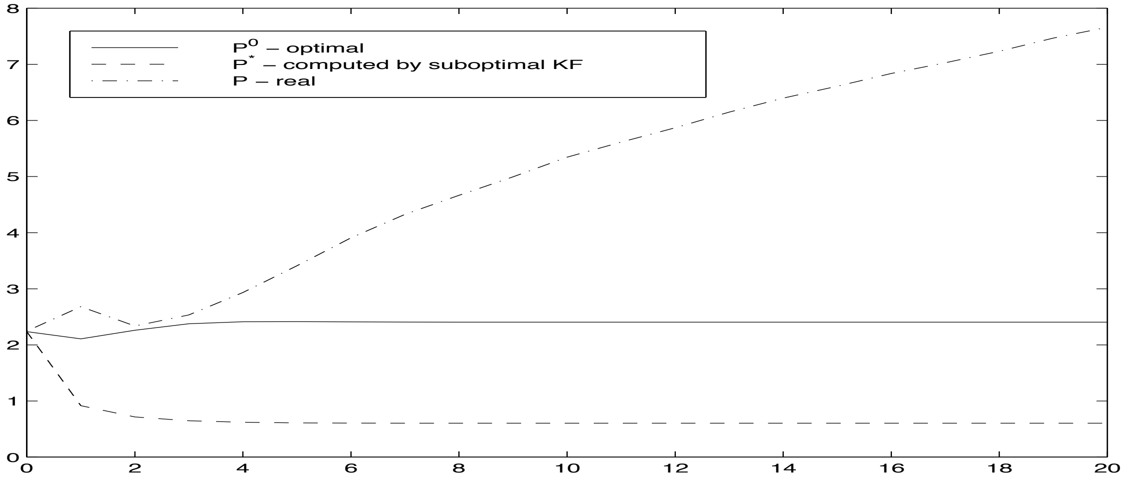

Figure 8.4: Assessing suboptimal filter performance in estimating $x_1(k)$ for a filter based on an erroneous $\mathbf{\Phi}$ matrix. The *real* error standard deviation is shown together with the standard deviation computed by the suboptimal filter itself. The standard deviation for the *optimal* filter is as in Figure 6.3 and is given as a reference.

where the assumed air friction coefficient $f = 5$ provides "drag". For simplicity, we assume that $u(t) = 0$ for all t, but a noise term $\mathbf{w}(k-1)$ with covariance matrix $\mathbf{Q}$ is added to the *deterministic* version of the model. The filter is now assessed by producing a large number of realizations of the real state vector (without friction) and the associated measurements. The suboptimal filter processes these observation. For each simulation run, the state and the estimate from the suboptimal filter are compared and the error computed. Averaging over a large number of simulations allows us to estimate the mean and standard deviation of the estimation errors. The result is shown if Figure 8.4 which is based on averages over 5000 Monte Carlo simulations. Note that $R^* = 1$ and $\mathbf{Q}^* = 0.2\mathbf{I}$ as above.

Again the filter is overly optimistic compared to the optimal results but this time a serious situation occurs since the real standard deviation diverges leading to estimates that are effectively useless. You may wonder if the estimates are still unbiased (they may not). Play with the *Matlab* code and investigate this.

8.4 Problems

Exercise 8.1 (Proof of Lemma 8.1)
(a) Define $\mathbf{D} = \mathbf{A} + \mathbf{B}\mathbf{C}^T$. Show that

$$\mathbf{A}^{-1} = \mathbf{D}^{-1} + \mathbf{D}^{-1}\mathbf{B}\mathbf{C}^T\mathbf{A}^{-1}$$

(Hint: Write $\mathbf{I}_n = \mathbf{D}^{-1}\mathbf{D}$ and insert for $\mathbf{D}$. Then multiply by $\mathbf{A}^{-1}$.)

(b) Based on the result in (a), show that

$$\mathbf{D}^{-1}\mathbf{B} = \mathbf{A}^{-1}\mathbf{B}(\mathbf{I}_n + \mathbf{C}^T\mathbf{A}^{-1}\mathbf{B})^{-1}$$

and use this to show part (a) of Lemma 8.1.

(c) Show part (b) of Lemma 8.1.

(Hint: Multiply the expression in (b) by $\mathbf{C}^T\mathbf{A}^{-1}$ from the right and rearrange terms.)

Exercise 8.2 (Proof of information filter)
We will in this exercise show equations (8.1b) and (8.2). The remaining equations in the information filter may be proved similarly [1].

(a) Use (6.13) to show that

$$\begin{aligned}\mathbf{P}(k|k) =&\mathbf{P}(k|k-1) - \mathbf{P}(k|k-1)\mathbf{H}(k)^T\times\\ &\{\mathbf{H}(k)\mathbf{P}(k|k-1)\mathbf{H}(k)^T + \mathbf{R}(k)\}^{-1}\mathbf{H}(k)\mathbf{P}(k|k-1)\end{aligned}$$

(b) Rearrange the expression in (a) further to show that

$$\begin{aligned}\mathbf{P}(k|k) =&\mathbf{P}(k|k-1) - \mathbf{P}(k|k-1)\mathbf{H}(k)^T\mathbf{R}(k)^{-1}\times\\ &\{\mathbf{H}(k)\mathbf{P}(k|k-1)\mathbf{H}(k)^T\mathbf{R}(k)^{-1} + \mathbf{I}\}^{-1}\mathbf{H}(k)\mathbf{P}(k|k-1)\end{aligned}$$

(c) Based on the result in (b), show (8.1b).

(Hint: Apply Lemma 8.1 (b) using $\mathbf{A} = \mathbf{P}(k|k-1)^{-1}$, $\mathbf{B} = \mathbf{H}(k)^T\mathbf{R}(k)^{-1}$, and $\mathbf{C} = \mathbf{H}(k)^T$.)

(d) By using the same trick as in (a), (b), show that

$$\mathbf{K}(k) =\mathbf{P}(k|k-1)\mathbf{H}(k)^T\mathbf{R}(k)^{-1}(\mathbf{H}(k)\mathbf{P}(k|k-1)\mathbf{H}(k)^T\mathbf{R}(k)^{-1} + \mathbf{I})^{-1}$$

(e) Use now part (a) of Lemma 8.1 with $\mathbf{A}, \mathbf{B}$ and $\mathbf{C}$ as in (c) to show that

$$\mathbf{K}(k) =(\mathbf{P}(k|k-1)^{-1} + \mathbf{H}(k)^T\mathbf{R}(k)^{-1}\mathbf{H}(k))^{-1}\mathbf{H}(k)^T\mathbf{R}(k)^{-1}.$$

(f) Use (e) and (8.1b) to show (8.2).

Exercise 8.3 (Better estimates with more observations)

(a) From the equations in the information filter, we found that $\mathbf{P}(k|k-1) - \mathbf{P}(k|k)$ is always positive semidefinite. Use this to show that

$$\text{var}[\mathbf{a}^T\hat{\mathbf{x}}(k|k)] \leq \text{var}[\mathbf{a}^T\hat{\mathbf{x}}(k|k-1)]$$

where $\mathbf{a}$ is a vector of length n. Interpret the result.

(b) Use now Theorem 5.1 to show the result in (a) directly. Based on this, show then that $\mathbf{P}(k|k-1) - \mathbf{P}(k|k)$ must be positive semidefinite.

Exercise 8.4 (Divergence phenomenon)
This exercise illustrates the *divergence phenomenon*, which occurs when either system noise or measurement noise or both are small. In such cases the Kalman filter may give completely wrong results. Consider the simple state space model

$$\begin{aligned} x(k) &= x(k-1) + b, \\ z(k) &= x(k) + v(k), \end{aligned}$$

where b is a very small *bias*, so small that, when we design our Kalman filter, we choose to neglect it. The noise term $v(k)$ is assumed to be Gaussian with expectation zero and variance equal to R. Our Kalman filter is based on the following (approximative) model (equal to the iid variables Example 4 in Section 6.4)

$$\begin{aligned} x^*(k) &= x^*(k-1), \\ z(k) &= x^*(k) + v(k). \end{aligned}$$

(a) Show that for the approximative model, the Kalman gain is given by

$$K(k) = \frac{P(0|0)}{kP(0|0) + R}.$$

What happens with $K * k)$ as $k \to \infty$?

(b) What happens with the estimate $\hat{x}^*(k|k)$ as k increases?

(Hint: Consider the difference $\hat{x}^*(k|k) - \hat{x}^*(k-1|k-1)$.)

(c) Compute $P^*(k|k)$, the covariance matrix for the error terms in the approximative model. Which impression does the approximative model give on the errors involved as $k \to \infty$?

(d) The Kalman filter only computes the error variance in the approximate model. The true error,

$$\tilde{e}(k|k) = \hat{x}^*(k|k) - x(k)$$

may however have an error of a quite different magnitude. Show that

$$\begin{aligned} \tilde{e}(k|k) =& \frac{R}{kP(0|0) + R}\tilde{e}(0|0) - \frac{P(0|0)}{kP(0|0) + R}\sum_{i=1}^{k} v(i) \\ &+ \frac{((k-1)k/2)P(0|0) + kR}{kP(0|0) + R}b. \end{aligned}$$

What happens with $\tilde{e}(k|k)$ as $k \to \infty$?

(e) Simulate the estimation error for your own choice of parameters.

(f) Summarize the results from this exercise to some general conclusions.

Exercise 8.5 (Suboptimal filtering)
Consider the same model as in Exercise 8.4. Put $R = 1$, $P(0|0) = 1$ and $b = 0.02$.

(a) Assume we know the constant external force b and we want to make an optimal filter based on this. Compute and plot the error variance for this filter.

(Hint: Use the filter given in Section 6.6.)

(b) Assume still you know b, but we believe that $R = 4$ and we therefore are considering a *suboptimal* filter. Compute and plot the error variance in the suboptimal filter both for the *apparent* error (the one we get if we believe in the wrong error) and the *real* error.

(c) Assume now that we only know about the appearance of b, but we do not know about its value. We still believe that $R = 4$. Construct a suboptimal filter where the error does not diverge by *introducing fictive system noise*. Compute and plot the apparent error variance for this filter.

(d) Generate a large number of realizations of the real estimation error and find its empirical distribution for some k's. Compare with the results from (a)-(c).

Exercise 8.6 (Misspecified covariance matrice)
Consider the example in Section 8.3.2. Compute $\mathbf{P}$ and $\mathbf{P}^*$ for $R^* = r$ and $\mathbf{Q}^* = 0.2r\mathbf{I}$ for different values of r. Discuss the results.

Chapter 9

Prediction and smoothing

So far we have been concentrating on "on-line" estimation, that is estimating $\mathbf{x}(k)$ given $\mathbf{z}(1),...,\mathbf{z}(k)$. Estimation of $\mathbf{x}(k)$ given the observations $\mathbf{z}(1),...,\mathbf{z}(l)$ where $l \neq k$ is however of interest in many cases. We have only seen one such case, where $l = k-1$, resulting in one step a head prediction, for which solutions are given directly from the Kalman filter. Prediction further into the future will be discussed in Section 9.1.

In many cases, interest is on estimating $\mathbf{x}(k)$ given data succeeding time point k (or t_k). Such types of estimation is usually called *smoothing*. For example, if there is no need in doing the estimation "on-line", the data analysis can be performed after, say, $\mathbf{z}(1),...,\mathbf{z}(N)$ is observed. Estimation of $\mathbf{x}(k)$ based on *all* data will then give more reliable results than only basing the estimation on the data up to time t. In fact, Theorem 5.1 in Section 5.1 shows that the optimal estimate for $\mathbf{x}(k)$ given that $\mathbf{z}(1),...,\mathbf{z}(N)$ is available is

$$\hat{\mathbf{x}}(k|N) = E[\mathbf{x}(k)|\mathbf{z}(1),...,\mathbf{z}(N)].$$

Of course, "on-line" calculation of these expectations is not possible, since it is necessary to have observed the later observations. However, the actual calculations of the estimators can be performed in a very similar manner to the Kalman filter. Note first that $\hat{\mathbf{x}}(N|N)$ is given directly from the Kalman filter at time N. But given $\hat{\mathbf{x}}(N|N)$ we can find an updating equation for $\hat{\mathbf{x}}(N-1|N)$. This can be repeated for finding $\hat{\mathbf{x}}(N-2|N)$ and so on, making the calculation of $\hat{\mathbf{x}}(k|N)$ possible by a *backwards* recursion from N to 1. Finding $\hat{\mathbf{x}}(k|N)$ for $k = 1,...,N$ is called the **fixed-interval smoother**.

In other cases, we are perhaps more interested in $\mathbf{x}(k)$ at a specific time, and we want to update our estimate for each time a new observation is collected. That is, we want to calculate $\hat{\mathbf{x}}(k|j)$ for $j = k, k+1, k+2, ...$ Also these estimates are available through recursive calculations, but in this case the calculations *is* possible to perform on-line. Such an approach is called **fixed-point smoothers**. Note that in principle, we could perform such estimation by using the fixed-interval smoother for $N = j$. The amount of calculation involved will however be huge, making this approach less attractive.

The last kind of smoothers we will be considering is the **fixed-lag** smoothers. In this case, we are interested in estimating $\mathbf{x}(k)$ given the data up to $k+L$, where L is a fixed number. Such estimates could be important if for instance the real interest is in $\hat{\mathbf{x}}(k|N)$, but where it is reasonable to assume that observations more than L time-steps ahead have negligible influence on $\mathbf{x}(k)$ given the data up to time $k+L$. Such estimates can then be calculated at time $k+L$ instead of waiting for all observations to be collected.

We will start in section 9.2 with a description on how to perform fixed-point smoothing since this is the easiest part. Section 9.3 consider fixed-interval smoothing while Section 9.4 deals with fixed-lag smoothing. Section 9.5 closes the chapter with some examples.

9.1 Predicting future

In many cases we are interested in predicting the future $\mathbf{x}(k+j)$ based on the observations $\mathbf{z}(1), ..., \mathbf{z}(k)$. From the Kalman filter algorithm, prediction of the system vector one step ahead is directly obtainable. Predicting the state further time steps apart is just as easy to obtain. There are many alternative ways of doing this, but perhaps the easiest approach is the one based on utilizing the already derived Kalman filter and the flexibility in our state space modeling. When prediction of $\mathbf{x}(k+j)$ is to be based on $\mathbf{z}(1), ..., \mathbf{z}(k)$, we may consider the observations $\mathbf{z}(k+1), ..., \mathbf{z}(k+j)$ as *missing*. In Section 4.3 we saw that missing data can be incorporated into the model by defining $\mathbf{H}^*(l) = \mathbf{0}$ for $l = k+1, ..., k+j$. Running the Kalman filter up to time point $k+j$ by this modified model, the predictions

$$\hat{\mathbf{x}}(k+j|k) = E[\mathbf{x}(k+j)|\vec{\mathbf{z}}(k)]$$

and the covariance matrix for the corresponding prediction errors

$$\mathbf{e}(k+j|k) = \hat{\mathbf{x}}(k+j|k) - \mathbf{x}(k+j)$$

are directly obtained.

Example 1 (A moving body, cont.)
Consider again the moving body example, and assume we want to predict the position for $k+j = 22, 23, ...$ based on the first $k = 21$ (simulated) observations shown in Figure 6.2. The results are shown in Figure 9.1. The upper panel shows the position, while the lower panel shows the velocity. In both plots, the solid line shows the estimate $x_i(l|l)$ for $l = 1, ..., 21$ and $x_i(l|21)$ for $l = 22, ..., 51$. The dashed lines shows the standard deviations for the error added and subtracted from the estimates. Note (for $l > 21$) that the errors increase as predictions are made further in time. □

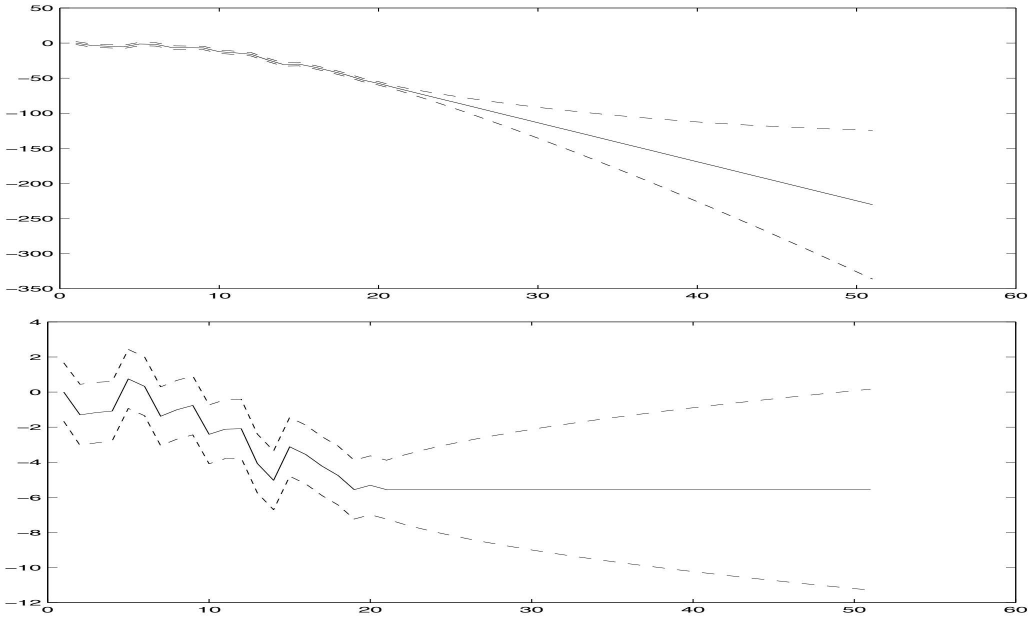

Figure 9.1: Prediction estimates (solid line) for $x_i(k+j|k)$ with $k=21$ and $j=1,...,30$. The dashed lines show $\hat{x}_i(k+j|k) \pm \sqrt{P(k+j|k)}$. For $l<k$, the ordinary Kalman filter estimates $\hat{\mathbf{x}}(l|l)) \pm \sqrt{P(l|l)}$ are shown. Upper panel shows the position while the lower panel shows the velocity of the moving body example. The computations are based on the (simulated) observations shown in Figure 6.2.

In time-series analysis, prediction of future *observations* is of particular interest. Now

$$\begin{aligned}\hat{\mathbf{z}}(k+j|k) =& E[\mathbf{z}(k+j)|\vec{\mathbf{z}}(k)]\\ =& E[\mathbf{H}(k+j)\mathbf{x}(k+j)+\mathbf{v}(k+j)|\vec{\mathbf{z}}(k)]\\ =& \mathbf{H}(k+j)\hat{\mathbf{x}}(k+j|k) \end{aligned}\tag{9.1}$$

and similar arguments result in

$$\mathrm{var}[\hat{\mathbf{z}}(k+j|k)-\mathbf{z}(k+j)] = \mathbf{H}(k+j)\mathbf{P}(k+j|k)\mathbf{H}(k+j)^T+\mathbf{R}(k+j). \tag{9.2}$$

These formulae can be easily implemented into our filter, making prediction of observations equally straightforward.

9.2 Fixed-point smoothing

Suppose we wish to estimate $\mathbf{x}(t)$ at some fixed point t based on data $\vec{\mathbf{z}}(k)$. For $k<t$, we are in an ordinary prediction situation, and the techniques in Section 9.1 applies. Our concern in this section will be situations where $k>t$. In particular, we would like to update our estimate of $\mathbf{x}(t)$ as new observations $\mathbf{z}(k), k = t+1, t+2, ...$ arrives. In the literature, special recursive equations are derived for performing this updating. An easier, although perhaps more heavy computational, approach is to utilize the flexibility of the already derived Kalman filter by modifying our state space model. In particular, we will, for $k > t$, add $\mathbf{x}(t)$ to the state vector. Denote

$$\mathbf{x}^*(k) = \begin{pmatrix} \mathbf{x}(t) \\ \mathbf{x}(k) \end{pmatrix}. \tag{9.3}$$

The system equation in this case becomes

$$\mathbf{x}^*(k) = \begin{pmatrix} \mathbf{I}_n & \mathbf{0} \\ \mathbf{0} & \mathbf{\Phi}(k-1) \end{pmatrix} \mathbf{x}^*(k-1) + \begin{pmatrix} \mathbf{0} \\ \mathbf{w}(k-1) \end{pmatrix}, \tag{9.4a}$$

while the observation equation becomes

$$\mathbf{z}(k) = \begin{pmatrix} \mathbf{0} & \mathbf{H}(k) \end{pmatrix} \mathbf{x}^*(k) + \mathbf{v}(k). \tag{9.4b}$$

Denote the estimation error covariance matrices resulting by running the Kalman filter on this modified state space model by $\mathbf{P}^*(k|k)$. Then $\hat{\mathbf{x}}(t|k)$ will be directly available as the first n components of $\hat{\mathbf{x}}^*(t|k)$, while $\mathbf{P}(t|k)$ is the upper left $n \times n$ submatrix of $\mathbf{P}^*(k|k)$.

9.3 Fixed-interval smoothing

The fixed-interval smoother is particularly important when estimation of parameters is of concern, see Chapter 10. In this case we assume $\mathbf{z}(k)$ are observed for all $k = 1, ..., N$, and we want to calculate the estimates $\hat{\mathbf{x}}(k|N)$ for all k. While in principle, modifications of the state space model could be constructed similar to the fixed-point smoother, the dimension of the state vector would have to be increased by n at each time-step, making the calculations impractical. A more efficient alternative in this case is to derive some special purpose recursive equations. Note that the estimate for $k = N$, $\hat{\mathbf{x}}(N|N)$ is directly available from the Kalman filter. We shall see that based on this, it is possible to derive an expression for $\hat{\mathbf{x}}(N-1|N)$, which again can be utilized for calculating $\hat{\mathbf{x}}(N-2|N)$ and so on. The calculation of the fixed-interval smoothers is therefore based on the Kalman filter, which is a forward recursive scheme and a new *backward* recursive scheme.

The following Theorem describes the backwards recursive equations, which we state without proof:

Theorem 9.1
The fixed interval smoother is obtained by the following (backwards) recursive equations:

$$\hat{\mathbf{x}}(k|N) = \hat{\mathbf{x}}(k|k) + \mathbf{A}(k)[\hat{\mathbf{x}}(k+1|N) - \hat{\mathbf{x}}(k+1|k)] \tag{9.5a}$$

$$\mathbf{P}(k|N) = \mathbf{P}(k|k) + \mathbf{A}(k)[\mathbf{P}(k+1|N) - \mathbf{P}(k+1|k)]\mathbf{A}^T(k) \tag{9.5b}$$

where $\mathbf{A}(k)$ *is given by*

$$\mathbf{A}(k) = \mathbf{P}(k|k)\mathbf{\Phi}^T(k-1)\mathbf{P}^{-1}(k+1|k). \tag{9.5c}$$

We see that all the quantities involved either are known from the Kalman filter *or* is given by calculations at time-point $k+1$, showing that we indeed have a backwards recursion algorithm for calculating the smoothers. Note that it is necessary to store $\hat{\mathbf{x}}(k|k)$ and $\mathbf{P}(k|k)$, but that the observations are not needed! Further, although both the prediction $\hat{\mathbf{x}}(k+1|k)$ and its error covariance matrix $\mathbf{P}(k+1|k)$ are needed in the equations above, they do not necessarly have to be stored, because they are easily calculated by (6.12a) and (6.13a).

9.4 Fixed-lag smoothing

For the fixed-lag smoother, we would like to estimate $\mathbf{x}(k)$ based on $\vec{\mathbf{z}}(k+L)$ for each k. The fixed-lag smoother is usually being used as an approximation to the fixed-interval smoother, the reason being that while the latter need all data to be collected before the estimates can be calculated, the former can be performed on-line (with a time-lag corresponding to the size of L).

The way to calculate these fixed-lag smoothers is similar to the approach for fixed-point smoothers. Also in this case, we will change our state space model by augmenting the state vector. Define in this case

$$\mathbf{x}^*(k) = \begin{pmatrix} \mathbf{x}(k) \\ \mathbf{x}_1(k) \\ \mathbf{x}_2(k) \\ \vdots \\ \mathbf{x}_L(k) \end{pmatrix} = \begin{pmatrix} \mathbf{x}(k) \\ \mathbf{x}(k-1) \\ \mathbf{x}(k-2) \\ \vdots \\ \mathbf{x}(k-L) \end{pmatrix} \tag{9.6}$$

where $\mathbf{x}_j(k) = \mathbf{x}(k-j)$ for $j = 1, ..., L$. Then, using the state model (4.6), we get

$$\mathbf{x}^*(k) = \begin{pmatrix} \mathbf{\Phi}(k-1) & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} \\ \mathbf{I}_n & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_n & \cdots & \mathbf{0} & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{I}_n & \mathbf{0} \end{pmatrix} \mathbf{x}^*(k-1) + \begin{pmatrix} \mathbf{I}_n \\ \mathbf{0} \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \end{pmatrix} \mathbf{w}(k) \tag{9.7a}$$

with the observation equation

$$\mathbf{z}(k) = \begin{pmatrix} \mathbf{H} & \mathbf{0} & \mathbf{0} & \ldots & \mathbf{0} \end{pmatrix} \mathbf{x}^*(k) + \mathbf{v}(k). \tag{9.7b}$$

Applying the Kalman filter on this augmented model (with size $(L+1)n$ on the state vector), we obtain not only the estimates $\hat{\mathbf{x}}(k|k)$ but also $\hat{\mathbf{x}}_i(k|k) = \hat{\mathbf{x}}(k-i|k), i = 1, 2, ..., L$. For $i = L$, the quantity of interest is obtained.

Note that we in this case need to specify the distribution for the initial state vector $\mathbf{x}^*(0) = (\mathbf{x}(0)^T, ..., \mathbf{x}(-L)^T)^T$ in the augmented model. Note further that $\hat{\mathbf{x}}(k|k+L)$ is the last element of $\hat{\mathbf{x}}^*(k+L|k+L)$, and similarly for $\mathbf{P}(k|k+L)$.

Calculating fixed-lag smoothers by this procedure has the advantage of utilizing the earlier derived Kalman filter. A disadvantage is however that the matrices involved can be quite large when L increases. There does exist other algorithms that are more efficient, although we will not this discuss this further here.

9.5 Example

In this section we will discuss the effect of smoothing on the random walk example.

Example 3 (Random walk, cont.)
Consider the random walk model

$$\begin{aligned} x(k) &= x(k-1) + w(k-1) \\ z(k) &= x(k) + v(k). \end{aligned}$$

Assume first we want to estimate $x(25)$ based on data $\vec{z}(k)$ for different k's. Now, for $k \leq 25$,

$$\begin{aligned} \hat{x}(25|k) =& \hat{x}(k|k), \\ P(25|k) =& P(k|k) + \sum_{l=k+1}^{25} Q(l-1) \end{aligned}$$

which is directly obtained by noticing that

$$x(25) = x(k) + \sum_{l=k+1}^{25} w(k).$$

Consider now the case $k > 25$. We then need to apply the method described in Section 9.2. First we augment our state variable to

$$\mathbf{x}^*(k) = \begin{pmatrix} x(25) \\ x(k) \end{pmatrix}.$$

giving the state space model

$$\begin{aligned} \mathbf{x}^*(k) &= \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \mathbf{x}^*(k-1) + \begin{pmatrix} 0 \\ w(k-1) \end{pmatrix}, \\ z(k) &= \begin{pmatrix} 0 & 1 \end{pmatrix} \begin{pmatrix} x(25) \\ x(k) \end{pmatrix} + v(k). \end{aligned}$$

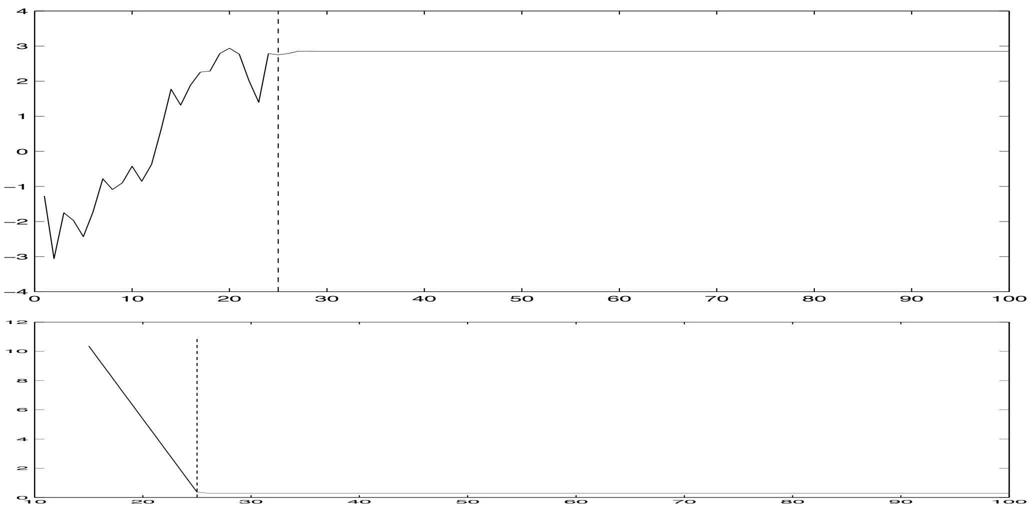

Figure 9.2: $\hat{x}(25|k)$ (upper panel) and $\sqrt{P(25|k)}$ (lower panel) as a function of k based on the simulated data given in Figure 4.1. In both plots, the vertical dashed line indicate time point 25.

By running the Kalman filter on this state space model, we obtain $\hat{x}(25|k)$ and $P(25|k)$ for $k > 25$.

In Figure 9.2, $\hat{x}(25|k)$ and $P(25|k)$ are plotted for $k = 15, ..., 100$ using the simulated data given in Figure 4.1. For $k < 25$, the variance is very high and the estimates depend heavily on the observations. For $k > 25$, only small corrections are made for the first proceeding observations, after which no changes are made. Similarly, the variance is stabilizing, reflecting that $z(k)$-observations for k much larger than 25 do not influence much on $\hat{x}(25|k)$.

Consider next fixed-interval smoothing. We then need to apply the backwards recursions given in Theorem 9.1. Figure 9.3 shows $\hat{x}(k|N)$ and $P(k|N)$ for $N = 100$ based on the same data as above. The upper panel shows both the smoothed estimates (solid line) and the filtered ones (dashed line). Note that the smoothed curve is in fact much smoother than the filtered one. This is a typical behavior, due to the use of succeeding points in the estimation. In the lower panel the error variances for the smoothed estimates (solid line) is compared to the error variances for the filtered estimates. Although both curves has the same type of shape, the smoothed one is lower than the filtered one, which again is due to that the smoothed estimates utilizes more of the data.

Consider finally fixed-lag smoothing. We assume we want to estimate $\hat{x}(k|k+L)$ with $L = 2$. In this case we need to augment the state vector to

$$\mathbf{x}^*(k) = \begin{pmatrix} x(k) \\ x(k-1) \\ x(k-2) \end{pmatrix}$$

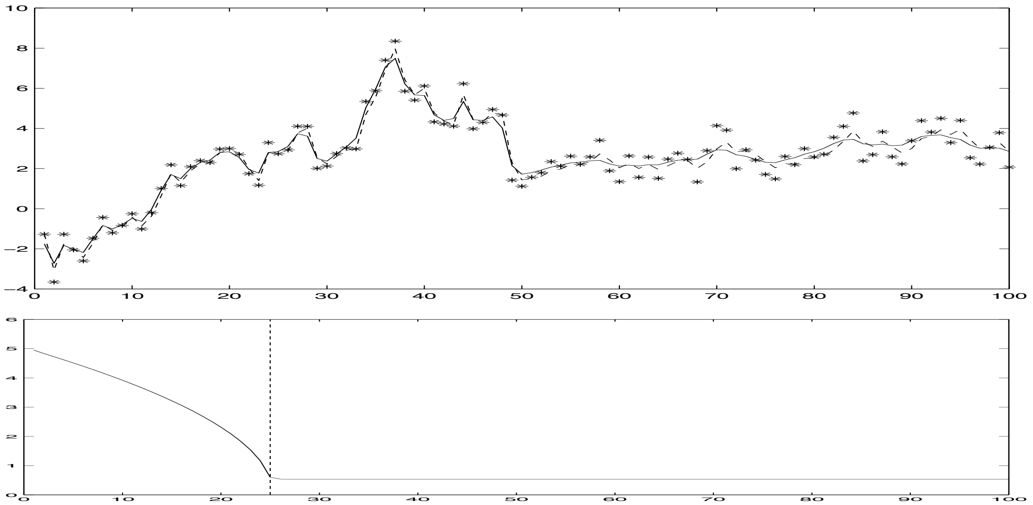

Figure 9.3: $\hat{x}(k|N)$ (upper panel) and $\sqrt{P(k|N)}$ (lower panel) as a function of k based on the simulated data given in Figure 4.1. In both plots, the corresponding filtered estimates are shown as dashed lines.

giving the state space model

$$\begin{aligned} x^*(k) &= \begin{pmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} \mathbf{x}^*(k-1) + \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} w(k) \\ z(k) &= \begin{pmatrix} 1 & 0 & 0 \end{pmatrix} \mathbf{x}^*(k) + v(k). \end{aligned}$$

Figure 9.4 compares the results from using the fixed-lag smoothing (solid lines) with the results using fixed-interval smoothing (dashed lines). The upper panel shows the estimates, while the lower panel shows the error variances. Note that for $k < 50$, the results are almost identical, showing that, given $\vec{z}(k+2)$, later observations give no extra information about $x(k)$. For $k > 50$, there are some small differences between the two smoothers. The reason for this is that in this case there is a larger dependence between the states at different time points (the signal to noise ratio is larger). This implies that observations taken further apart from the time point of interest will contain more information, making the need for choosing a larger L.

□

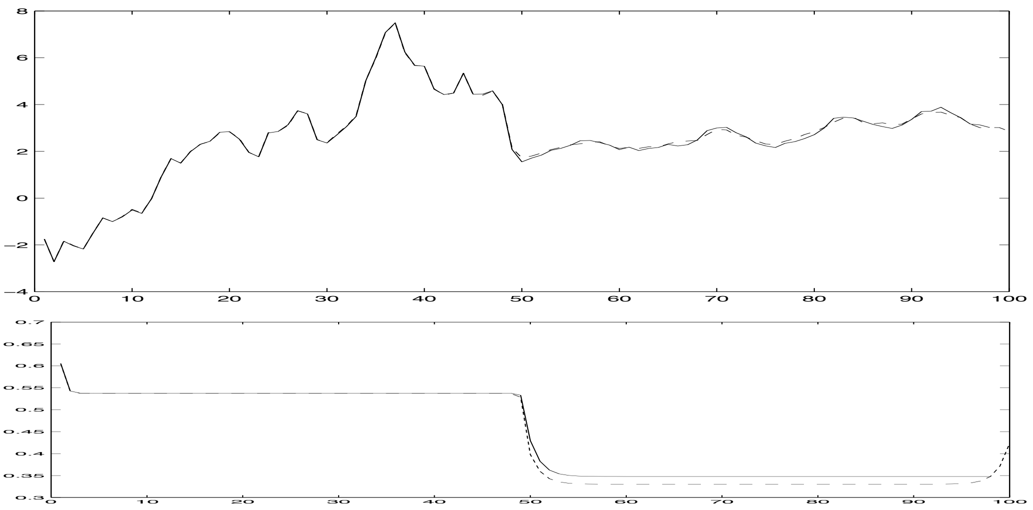

Figure 9.4: For the simulated data given in Figure 4.1, $\hat{x}(k|k+2)$ (solid line) and $\hat{x}(k|N)$ (dashed line) is shown in the upper panel, $\sqrt{P(k|k+2)}$ (solid line) and $\sqrt{P(k|N)}$ is shown in the lower panel.

9.6 Problems

Exercise 9.1 (Predicting future in time-invariant models)

(a) Consider a time-invariant state space model. Show that

$$\mathbf{x}(k+j) = \mathbf{\Phi}^j\mathbf{x}(k) + \sum_{l=0}^{j-1}\mathbf{\Phi}^{j-l-1}\mathbf{w}(k+l).$$

(b) Show that

$$\hat{\mathbf{x}}(k+j|k) = E[\mathbf{x}(k+j)|\vec{\mathbf{z}}(k)] = \mathbf{\Phi}^j\hat{\mathbf{x}}(k|k).$$

(Hint: Use the double expectation rule, conditioning on $\mathbf{x}(k)$.)

(c) Show that the error term $\mathbf{e}(k+j|k) = \hat{\mathbf{x}}(k+j|k) - \mathbf{x}(k+j)$ is given by

$$\mathbf{e}(k+j|k) = \mathbf{\Phi}^j[\hat{\mathbf{x}}(k|k) - \mathbf{x}(k)] - \sum_{l=0}^{j-1}\mathbf{\Phi}^{j-l-l}\mathbf{w}(k+l).$$

(d) Use again the double expectation rule to show that the prediction error covariance matrix $\mathbf{P}(k+j|k) = \mathrm{VAR}[\mathbf{e}(k+j|k)]$ is given by

$$\mathbf{P}(k+j|k) = \mathbf{\Phi}^j\mathbf{P}(k|k)(\mathbf{\Phi}^j)^T + \sum_{l=0}^{j-1}\mathbf{\Phi}^{j-l-1}\mathbf{Q}(\mathbf{\Phi}^{j-l-1})^T.$$

(*e*) Consider the univariate model

$$
\begin{aligned}
x(k) &= ax(k-1) + w(k-1) \\
z(k) &= x(k) + v(k)
\end{aligned}
$$

where $\{w(k)\}$ and $\{v(k)\}$ are white noise processes with zero expectations and $\text{var}[w(k)] = Q$, $\text{var}[v(k) = R$. What happens with $P(k+j|k)$ when $j \to \infty$? Discuss for different values of a.

(*f*) Implement the procedure for prediction in Section 9.1 for the model in (*e*) and check the conclusions you made in (*e*) with the ones obtained with the prediction filter.

Exercise 9.2 (A moving body, cont.)
Consider our two-dimensional moving body model (the discrete version given by eqns. (6.24)).

(*a*) Simulate $N = 50$ observations from the model with your own choice of parameters.

(*b*) Run the Kalman filter to obtain $\hat{\mathbf{x}}(k|k)$ and $\mathbf{P}(k|k)$ for $k = 1, ..., 50$.

(*c*) Run the fixed interval smoother for obtaining $\hat{\mathbf{x}}(k|N)$ and $\mathbf{P}(k|N)$. Compare with your results in (*b*).

(*d*) Run the fixed lag smoother with different values of j. Compare with the results in (*c*).

(*e*) Assume you are particular interested in the time point $k = 25$. Run the fixed point smoother in combination with the prediction filter to obtain estimates of $\mathbf{x}(25)$ at all time points $1, ..., 50$.

(*f*) Repeat (*a*) – (*e*) for different values on the variances for the noise. Try to make some general conclusions.

Exercise 9.3 (A simple hydraulic model)
We will in this exercise consider a simple hydraulic model for the river "Overflow". The purpose of the model is to use it for prediction of possible overflow situations on the lowest part of the watercourse. We then need a Kalman filter and observations from the river. The rainfall area is modeled as a series of n reservoirs, each representing partial rainfall areas. The water level in reservoir j at time point t is $x_j(t)$ and the water flow out of reservoir is $a_j x_j(t)$. (Such a reservoir is called linear.). Reservoir number j ($j > 1$) receive water from the above reservoir $j - 1$ in addition to water

fall $u_j(t) + w_j(t)$ which falls down in the sub-area. We then have the following set of linear differential equations:

$$\begin{aligned}
\dot{x}_1(t) &= -a_1 x_1(t) + u_1(t) + w_1(t), \\
\dot{x}_j(t) &= -a_j x_j(t) + a_{j-1} x_{j-1}(t) + u_j(t) + w_j(t), \quad 1 < j \leq n.
\end{aligned}$$

In order to get a manageable problem, we will assume that $n = 4$. All volumes are measured in millimeter (mm) water distributed over the sub-area and all water flows (rates) in mm/day. (NB! One millimeter doesn't sound much, but it becomes a lot if the sub-area is a few square kilometers.) The constants $a_1, \ldots, a_4$ then are on the scale 1/day. Use $a_1 = 0.2$, $a_2 = 0.7$, $a_3 = 0.2$ and $a_4 = 0.4$.

(a) Put the system of differential equations in state space form. Discretize the model with sampling time equal to 1 day. (That is, find the matrices $\boldsymbol{\Phi}$ and $\boldsymbol{\Psi}$ in *Matlab* .) We assume the rainfall is constant over a day.

(b) Assume now we measure the water flow out of reservoir 4. Show that the system then is observable. Why is this not the case if one of the other water flows are measured?

(c) Consider now u_j for $j = 1, \ldots, 4$ as external forces on the model. Show that the system is reachable.

The initial amounts of water in each reservoir are random variables with mean vector and covariance matrix

$$\mu = \begin{pmatrix} 100 \\ 60 \\ 300 \\ 200 \end{pmatrix}, \quad \mathbf{P} = \begin{pmatrix} 10^2 & 0 & 0 & 0 \\ 0 & 15^2 & 0 & 0 \\ 0 & 0 & 20^2 & 0 \\ 0 & 0 & 0 & 15^2 \end{pmatrix}.$$

The covariance matrix is diagonal, which is not the case in reality. (Why?) The water falls have known mean, so we model this mean as an external force $u_j(k)$, $j = 1, \ldots, 4$. We assume

$$u_j(k) = 5 \sin\left(\frac{2\pi(k + 10j)}{365}\right) + 20\,\text{mm/day}.$$

The random component in the water fall is modeled as system noise $w_j(k)$. We further assume that $\{w_j(k)\}$ are independent variables (and independent of the initial state) with mean zero and covariance matrix

$$\mathbf{Q} = \begin{pmatrix} 8^2 & 0 & 0 & 0 \\ 0 & 7^2 & 0 & 0 \\ 0 & 0 & 5^2 & 0 \\ 0 & 0 & 0 & 2^2 \end{pmatrix}.$$

(*d*) Calculate the mean and standard deviation for all water levels over 1 year (without filtering). Why does the covariance matrix for the state vector approach a constant matrix? Why is this constant matrix independent of the initial covariance matrix?

(*e*) The Kalman filter is based on daily measures of the water flow out from the lowest reservoir. The measurement errors are independent and Gaussian distributed with expectation zero and standard deviation equal to 2 mm/day. Calculate the standard deviations for the errors in the filtered estimates over 1 year.

(*f*) After all measurements are collected, it may be of interest to know as exactly as possible the water level in the year that has passed. Calculate therefore the smoothed solution, and plot the standard deviation for the error of the smoothed estimate.

(*g*) We want to be able to predict the water level at time point $k + L$ based on observations up to time point k. How large can L be made so that the standard deviation for the prediction error in the water level is smaller than 95 percent of the standard error based on only using average values?

Chapter 10

Filter adaption

Theoretical arguments rarely determine all the coefficients and covariance matrices $\{\mathbf{\Phi}(k)\}$, $\{\mathbf{H}(k)\}$, $\{\mathbf{Q}(k)\}$ and $\{\mathbf{R}(k)\}$ influencing the filter. Those that remain unknown are usually found statistically, that is by fitting observed data. Let θ be the vector of unknown parameters, and $\mathbf{D}$ the data. The idea is to adjust θ so as to be compatible with $\mathbf{D}$. Many ways of doing this have been proposed for linear state space models. We shall here only consider so-called *maximum likelihood estimation*, which can be formulated very simply and in a way that applies to all state space models simultaneously. Note that we are in this case considering a different estimation problem compared to the one discussed in Chapter 5. In this chapter the quantity θ is assumed to be a *fixed* parameter, in contrast to the situation in Chapter 5 where a *random* variable was to be estimated. The difference require separate estimation methods.

Maximum likelihood estimation is the most widely used fitting technique in statistics in general with a number of good theoretical properties. One such property is that as the number of observations in $\mathbf{D}$ becomes infinite, then the estimate $\hat{\theta}$ of θ will tend to θ itself. This is known as **consistency**. To give a mathematical definition, let θ_i be the i'th unknown parameter of the state space model and $\hat{\theta}_i$ its estimate, obtained from $\mathbf{D}$ in some way. The estimation *error* is then $\hat{\theta}_i - \theta_i$. Note that error varies randomly with both the system and measurement noise that have influenced the actual observations $\mathbf{D}$. We might consider the *expected* size of the error, for example

$$E[|\hat{\theta}_i - \theta_i|] \qquad \text{or} \qquad E[(\hat{\theta}_i - \theta_i)^2].$$

Consistency means that either of these quantities converges to 0 as the length of the $\mathbf{D}$-vector tends to infinity. The quantity on the right can be decomposed according to

$$E[(\hat{\theta}_i - \theta_i)^2] = \{b(\hat{\theta}_i)\} + \mathrm{var}[\hat{\theta}_i],$$

where $b(\hat{\theta}_i) = E[\hat{\theta}_i - \theta_i]^2$. This is known as the **brutto variance** formula and shows that the expected squared error on the left is a sum of a **bias** term and the variance of $\hat{\theta}_i$. The former expresses whether there is a tendency for $\hat{\theta}_i$ to underestimate or overestimate the true θ_i. Usually, with state space models, the variance term is the dominating one. One might wish to carry out the estimation so that $E[\hat{\theta}_i] = \theta_i$ exactly, and hence $b(\hat{\theta}_i) = 0$. The variance is then the only contributer to the expected squared error $E[(\hat{\theta}_i - \theta_i)^2]$. This situation is known as the **unbiased** one, and $\hat{\theta}_i$ is an **unbiased** estimate of θ_i. With the state space models it is often hard to obtain such unbiased estimates.

One way of constructing an estimate $\hat{\theta}$ of the vector θ of unknown parameters, is to reason as follows: The actual data $\mathbf{D}$ have been generated from the unknown θ according to mechanisms that are partially random. This applies both to the realization $\vec{\mathbf{x}}(N)$ of the system process and the realization $\vec{\mathbf{z}}(N)$ of the measurement process. Often we only have $\vec{\mathbf{z}}(N)$ and our data is then $\mathbf{D} = \vec{\mathbf{z}}(N)$. If $\vec{\mathbf{x}}(N)$ has been observed as well, then $\mathbf{D} = (\vec{\mathbf{x}}(N), \mathbf{D} = \vec{\mathbf{z}}(N))$. We may in either case consider the **likelihood function**

$$L_{\mathbf{D}}(\theta) = p(\mathbf{D}|\theta) \tag{10.1}$$

where $p(\mathbf{D}|\theta)$ is the joint probability density function of $\mathbf{D}$. How the function $L_{\mathbf{D}}(\theta)$ may be computed for given θ, is the main topic of this chapter. But for the moment the point is that *different* θ's produce different evaluations of how likely the observed sequence $\mathbf{D}$ is. Suppose, for example, that θ^* is far from the true θ. Then $L_{\mathbf{D}}(\theta^*)$ might well be rather low, because the actual observations have been generated under a very different mechanism and are therefore unlikely to occur under θ^*. Carrying this idea further we may vary θ^* systematically and compute $L_{\mathbf{D}}(\theta^*)$ everywhere. This traces out a surface $L_{\mathbf{D}}(\theta)^*$ over the whole θ-space. The surface is likely to be at its largest *when θ^* is close to the true θ*. A maximum likelihood estimate $\hat{\theta}_{ML}$ is a *maximum point*, i.e.

$$L_{\mathbf{D}}(\hat{\theta}_{ML}) = \sup_{\theta^*} L_{\mathbf{D}}(\theta^*). \tag{10.2}$$

Note that the argument behind (10.2) is completely general and has nothing to do with state space models as such. However, to find $\hat{\theta}_{ML}$ in practice, we have to be able to compute (10.1) and then optimize according to (10.2). This raises several issues. The first is that $L_{\mathbf{D}}(\theta) = p(\mathbf{D}|\theta)$ is not even defined unless the probability distribution of the system and measurement noise is known. In principle we need the probability distribution of all noise terms jointly, but in practice we have already made radical simplifications in that noise processes were assumed to be white and uncorrelated. In addition, we have to specify the shape of the probability distribution of each $\mathbf{v}(k)$ and $\mathbf{w}(k)$, see (4.6). It is usual to proceed as if they were Gaussian. The validity of such an assumption is often hard to verify. Neither should it be taken too literally. It does not mean that the filter adaption method we are going to present can not be used for non-Gaussian noise, but it indicates that the method might work best when noise distributions are not too far form Gaussian ones.

There is a big technical advantage in computing $L_{\mathbf{D}}(\theta)$ on the basis of Gaussianity. It then turns out that $L_{\mathbf{D}}(\theta)$ has a simple relationship to the Kalman filter, even when $\vec{\mathbf{x}}(N)$ is not observed, see Section 10.2. To find $\hat{\theta}_{ML}$ we may simply invoke the Kalman filter to compute $L_{\mathbf{D}}(\theta)$ and then apply a general numerical optimization method from a program library such as *Matlab* to maximize $L_{\mathbf{D}}(\theta)$. The only technical problem with this approach is to make the optimizer converge to the maximum of the $L_{\mathbf{D}}(\theta)$-surface. This might at times be a serious practical obstacle, and we therefore present (Section 10.3) a special procedure, known as the EM algorithm, that is bound to produce at least a *local* maximum on the $L_{\mathbf{D}}(\theta)$-surface[1]. Note that for this method the system process $\vec{\mathbf{x}}(N)$ is estimated *jointly* with θ. Occasionally $\vec{\mathbf{x}}(N)$ has at a training stage been observed jointly with $\vec{\mathbf{z}}(N)$. The problem of adapting the filter is then simpler. We treat this situation first.

10.1 Observing the System process

Suppose $\{\mathbf{x}(k)\}, k = 0, ..., N$ and $\{\mathbf{z}(k)\}, k = 1, ..., N$ are observed. Their joint density can be factorized as

$$p(\vec{\mathbf{x}}(N), \vec{\mathbf{z}}(N)|\theta) = p(\vec{\mathbf{x}}(N)|\theta)p(\vec{\mathbf{z}}(N)|\vec{\mathbf{x}}(N);\theta). \tag{10.3}$$

Note that we have included $\mathbf{x}(0)$ in the definition of $\vec{\mathbf{x}}(N)$. This might not be true in practice, but simplifies a technical detail. After mastering the contents of this chapter, the reader will be able to produce the required modifications him/herself. Looking first at $p(\vec{\mathbf{x}}(N)|\theta)$, an elementary factorization gives

$$p(\vec{\mathbf{x}}(N)|\theta) = p(\mathbf{x}(0)|\theta)p(\mathbf{x}(1)|\mathbf{x}(0);\theta)p(\mathbf{x}(2)|\vec{\mathbf{x}}(1);\theta)\cdots p(\mathbf{x}(N)|\vec{\mathbf{x}}(N-1);\theta)$$

where all the densities on the right hand side is known directly from the model specifications (4.6a). In particular,

$$\begin{aligned} p(\mathbf{x}(0);\theta) &= \mathcal{N}(\mu(0), \mathbf{P}(0|0)), \\ p(\mathbf{x}(k)|\vec{\mathbf{x}}(k-1);\theta) &= \mathcal{N}(\mathbf{\Phi}(k-1)\mathbf{x}(k-1), \mathbf{Q}(k-1)). \end{aligned}$$

Similarly for $p(\vec{\mathbf{z}}|\vec{\mathbf{x}})$. In this case we write

$$\begin{aligned} &p(\vec{\mathbf{z}}(N)|\vec{\mathbf{x}}(N);\theta) \\ &\quad= p(\mathbf{z}(1)|\vec{\mathbf{x}}(N);\theta)p(\mathbf{z}(2)|\vec{\mathbf{x}}(N), \vec{\mathbf{z}}(1);\theta)\cdots p(\mathbf{z}(N)|\vec{\mathbf{x}}(N), \vec{\mathbf{z}}(N-1);\theta). \end{aligned}$$

The second factor in (10.3) is equally simple. Now

$$p(\mathbf{z}(k)|\vec{\mathbf{x}}(N), \vec{\mathbf{z}}(k-1);\theta) = \mathcal{N}(\mathbf{H}(k)\mathbf{x}(k), \mathbf{R}(k)).$$

[1] That θ^* is a *local* maximum means that $L_{\mathbf{D}}(\theta^*) \geq L_{\mathbf{D}}(\theta)$ for all θ in a neighborhood of θ^*. For a *global* maximum the inequality is valid everywhere.

Combining all these factors we obtain from (10.3) the likelihood function

$$\begin{aligned}L_{\mathbf{xz}}(\theta) =&\frac{1}{(2\pi)^{n/2}|\mathbf{P}(0|0)|^{1/2}}e^{-\frac{1}{2}(\mathbf{x}(0)-\mu(0))^T\mathbf{P}(0|0)^{-1}(\mathbf{x}(0)-\mu(0))}\times\\ &\prod_{k=1}^{N}\frac{1}{(2\pi)^{n/2}|\mathbf{Q}(k-1)|^{1/2}}e^{-\frac{1}{2}(\mathbf{x}(k)-\mathbf{\Phi}(k-1)\mathbf{x}(k-1))^T\mathbf{Q}(k-1)^{-1}(\mathbf{x}(k)-\mathbf{\Phi}(k-1)\mathbf{x}(k-1))}\times\\ &\prod_{k=1}^{N}\frac{1}{(2\pi)^{m/2}|\mathbf{R}(k)|^{1/2}}e^{-\frac{1}{2}(\mathbf{z}(k)-\mathbf{H}(k)\mathbf{x}(k))^T\mathbf{R}(k)^{-1}(\mathbf{z}(k)-\mathbf{H}(k)\mathbf{x}(k))}.\end{aligned} \tag{10.4}$$

which is to be maximized with respect to θ. It is always better to pass to logarithms because on the original scale $L_{\mathbf{xz}}(\theta)$ suffers from hopeless numerical underflow. A computer usually rounds it off as zero! Since the logarithm is a monotone function, the maximum point of the log-likelihood is equal to the maximum point of the likelihood function itself. Accordingly, consider

$$\begin{aligned}l_{\mathbf{xz}}(\theta) =&\log L_{\mathbf{xz}}(\theta)\\ =&-\frac{n}{2}\log(2\pi)-\frac{1}{2}\log|\mathbf{P}(0|0)|-\frac{1}{2}(\mathbf{x}(0)-\mu(0))^T\mathbf{P}(0|0)^{-1}(\mathbf{x}(0)-\mu(0))\\ &-\frac{Nn}{2}\log(2\pi)-\frac{1}{2}\sum_{k=1}^{N}\log|\mathbf{Q}(k-1)|\\ &-\frac{1}{2}\sum_{k=1}^{N}(\mathbf{x}(k)-\mathbf{\Phi}(k-1)\mathbf{x}(k-1))^T\mathbf{Q}(k-1)^{-1}(\mathbf{x}(k)-\mathbf{\Phi}(k-1)\mathbf{x}(k-1))\\ &-\frac{Nm}{2}\log(2\pi)-\frac{1}{2}\sum_{k=1}^{N}\log|\mathbf{R}(k)|\\ &-\frac{1}{2}\sum_{k=1}^{N}(\mathbf{z}(k)-\mathbf{H}(k)\mathbf{x}(k))^T\mathbf{R}(k)^{-1}(\mathbf{z}(k)-\mathbf{H}(k)\mathbf{x}(k)),\end{aligned} \tag{10.5}$$

which is to be maximized with respect to unknown parameters in $\{\mathbf{\Phi}(k)\}$, $\{\mathbf{H}(k)\}$, $\{\mathbf{Q}(k)\}$ and $\{\mathbf{R}(k)\}$ in order to obtain the maximum likelihood estimate for θ. As discussed in Section 4.3, there will be many cases in which nothing is known about the process before the data is observed. In that case the first line (involving $\mathbf{P}(0|0)$) will be reduced to a constant that can be neglected in the further analysis.

Example 3 (Random walk, cont.)
Consider the random walk model

$$\begin{aligned}x(k) &= x(k-1)+w(k-1)\\ z(k) &= x(k)+v(k).\end{aligned}$$

Assume that $Q(k-1) = Q$ and $R(k) = R$ for all k. Then Q and R are the two unknown parameters and $\theta = (Q, R)$. Suppose $P(0|0) = \infty$. Then the log-likelihood function (10.5) simplifies to

$$\begin{aligned} l_{xz}(Q,R) = &-\frac{N}{2}\log(2\pi) - \frac{N}{2}\log(Q) - \frac{1}{2Q}\sum_{k=1}^{N}(x(k)-x(k-1))^2 \\ &-\frac{N}{2}\log(2\pi) - \frac{N}{2}\log(R) - \frac{1}{2R}\sum_{k=1}^{N}(z(k)-x(k))^2. \end{aligned}$$

which is a rare case where the maximum can be found analytically by equating. Firstly

$$\frac{\partial}{\partial Q} l_{xz}(Q,R) = -\frac{N}{2Q} + \frac{1}{2Q^2}\sum_{k=1}^{N}(x(k)-x(k-1))^2$$

which yields

$$\widehat{Q}_{ML} = \frac{1}{N}\sum_{k=2}^{N}(x(k)-x(k-1))^2. \tag{10.6}$$

as the maximum likelihood estimate of Q. In the same manner

$$\frac{\partial}{\partial R} l_{xz}(Q,R) = -\frac{N}{2R} + \frac{1}{2R^2}\sum_{k=1}^{N}(z(k)-x(k))^2$$

so that

$$\widehat{R}_{ML} = \frac{1}{N}\sum_{k=1}^{N}(z(k)-x(k))^2. \tag{10.7}$$

From the first 50 (simulated) samples in Figure 4.1, we obtained

$$\begin{aligned} \widehat{Q} &= 1.34 \\ \widehat{R} &= 0.43 \end{aligned}$$

which should be compared with the true values $Q = 1.0$, $R = 0.5$. □

10.2 Main method

Suppose θ is to be determined from $\mathbf{z}(1), ..., \mathbf{z}(N)$ without direct knowledge of the underlying system process. The likelihood function (10.1) then becomes

$$L_{\mathbf{z}}(\theta) = p(\mathbf{z}(1), .., \mathbf{z}(N)|\theta).$$

Recall that the right hand side is the joint density function of the observations. It is easy to compute this quantity by means of the Kalman filter. Indeed, we shall show that the likelihood function is given by

$$\begin{aligned} l_{\mathbf{z}}(\theta) =& \log(L_{\mathbf{z}}(\theta)) \\ =& -\frac{Nm}{2}\log(2\pi) - \frac{1}{2}\sum_{k=1}^{N}\log|\mathbf{S}(k)| \\ & -\frac{1}{2}\sum_{k=1}^{N}(\mathbf{z}(k) - \mathbf{H}(k)\hat{\mathbf{x}}(k|k-1))^T\mathbf{S}(k)^{-1}(\mathbf{z}(k) - \mathbf{H}(k)\hat{\mathbf{x}}(k|k-1)), \end{aligned} \tag{10.8}$$

where $|\mathbf{S}(k)|$ is the determinant of the matrix $\mathbf{S}(k)$. Note that both $\hat{\mathbf{x}}(k|k-1)$ and $\mathbf{S}(k)$ in (10.8) are available form the Kalman filter recursion. To compute $l_{\mathbf{z}}(\theta)$ for given θ, we need to do no more than invoking the Kalman filter and adding the terms in (10.8) together. Subsequently the sum can be maximized with respect to θ through some numerical method in order to obtain the maximum likelihood estimate.

To prove (10.8) first recall the elementary factorization

$$\begin{aligned} p(&\mathbf{z}(1),\mathbf{z}(2),..,\mathbf{z}(N)|\theta) \\ &= p(\mathbf{z}(1)|\theta)p(\mathbf{z}(2)|\vec{\mathbf{z}}(1);\theta)p(\mathbf{z}(3)|\vec{\mathbf{z}}(2);\theta)\cdots p(\mathbf{z}(N)|\vec{\mathbf{z}}(N-1);\theta). \end{aligned} \tag{10.9}$$

The general term $p(\mathbf{z}(k)|\vec{\mathbf{z}}(k-1);\theta)$ is the density of $\mathbf{z}(k)$ given the history $\vec{\mathbf{z}}(k-1)$ of the observed series up to $k-1$. By (6.8)

$$\mathbf{z}(k) = \mathbf{H}(k)\hat{\mathbf{x}}(k|k-1) + \tilde{\mathbf{z}}(k|k-1),$$

where $\tilde{\mathbf{z}}(k|k-1)$ is the innovation of $\mathbf{z}(k)$ and $\hat{\mathbf{x}}(k|k-1) = E[\mathbf{x}(k)|\vec{\mathbf{z}}(k-1)]$ is a constant when $\vec{\mathbf{z}}(k-1)$ is fixed. Moreover, recall (Lemma 6.1 (6.9b)) that $\tilde{\mathbf{z}}(k|k-1)$ is stochastically independent of $\vec{\mathbf{z}}(k-1)$. It follows that $\mathbf{z}(k)$, in the conditional distribution given $\vec{\mathbf{z}}(k-1)$, has the same covariance matrix as $\tilde{\mathbf{z}}(k|k-1)$, which is $\mathbf{S}(k)$. The mean vector is $\mathbf{H}(k)\hat{\mathbf{x}}(k|k-1)$ and so

$$p(\mathbf{z}(k)|\vec{\mathbf{z}}(k-1);\theta) = \mathcal{N}(\mathbf{H}(k)\hat{\mathbf{x}}(k|k-1), \mathbf{S}(k)).$$

The logarithm of the conditional density is thus

$$\begin{aligned} \log p(&\mathbf{z}(k)|\vec{\mathbf{z}}(k-1);\theta) \\ &= -\frac{m}{2}\log(2\pi) - \frac{1}{2}\log|\mathbf{S}(k)| \\ &\quad -\frac{1}{2}(\mathbf{z}(k) - \mathbf{H}(k)\hat{\mathbf{x}}(k|k-1))^T\mathbf{S}(k)^{-1}(\mathbf{z}(k) - \mathbf{H}(k)\hat{\mathbf{x}}(k|k-1)), \end{aligned}$$

and when these terms are added together, (10.8) follows.

Example 4 (iid variables, cont.)
Consider again the model

$$\begin{aligned}x(k) &= x(k-1),\\ z(k) &= x(k) + v(k).\end{aligned}$$

The unknown quantity we want to estimate in this case is the variance in the observation noise R, so that $\theta = R$. In this case, we were able to obtain simple analytical expressions for $\hat{x}(k|k)$ and $P(k|k)$, given in (6.22) and (6.20), respectively. Using this, and assuming $P(0|0) = \infty$, we obtain

$$S(k) = \frac{R}{k-1} + R = \frac{k}{k-1}R.$$

The log-likelihood for $\vec{z}(N)$ then becomes[2]

$$\begin{aligned}l_{\mathbf{z}}(R) =& -\frac{N}{2}\log(2\pi) - \frac{1}{2}\sum_{k=1}^{N}\log\left(\frac{k}{k-1}R\right)\\ &- \frac{1}{2R}\sum_{k=1}^{N}\frac{k-1}{k}(z(k)-\bar{z}(k-1))^2\\ =& C - \frac{N}{2}\log R - \frac{1}{2R}\sum_{k=1}^{N}\frac{k-1}{k}(z(k)-\bar{z}(k-1))^2\end{aligned}$$

where C is a constant not depending on R. In order to maximize the log-likelihood with respect to R, we first calculate the derivative of the log-likelihood:

$$\frac{\partial}{\partial R}l_{\mathbf{z}}(R) = -\frac{N}{2}\frac{1}{R} + \frac{1}{2R^2}\sum_{k=1}^{N}\frac{k-1}{k}(z(k)-\bar{z}(k-1))^2$$

and putting the derivative to zero, we obtain the maximum likelihood estimate

$$\hat{R} = \frac{1}{N}\sum_{k=1}^{N}\frac{k-1}{k}(z(k)-\bar{z}(k-1))^2. \tag{10.10}$$

Some (lengthy?) calculations can be performed to show that this expression actually is equal to

$$\frac{1}{N}\sum_{k=1}^{N}(z(k)-\bar{z}(N))^2$$

which is the ordinary estimate for the variance. Equation (10.10) does however have the nice feature that it is possible to calculate it recursively, while the standard formulation needs $\bar{z}(N)$ to be calculated first. □

[2] Actually, there are some complications due to the fact that $S(1) = \infty$. There is however possible to carry out the calculations rigorous by introducing a finite $P(0|0)$ which in the end is being increased to infinity. The same result will however apply.

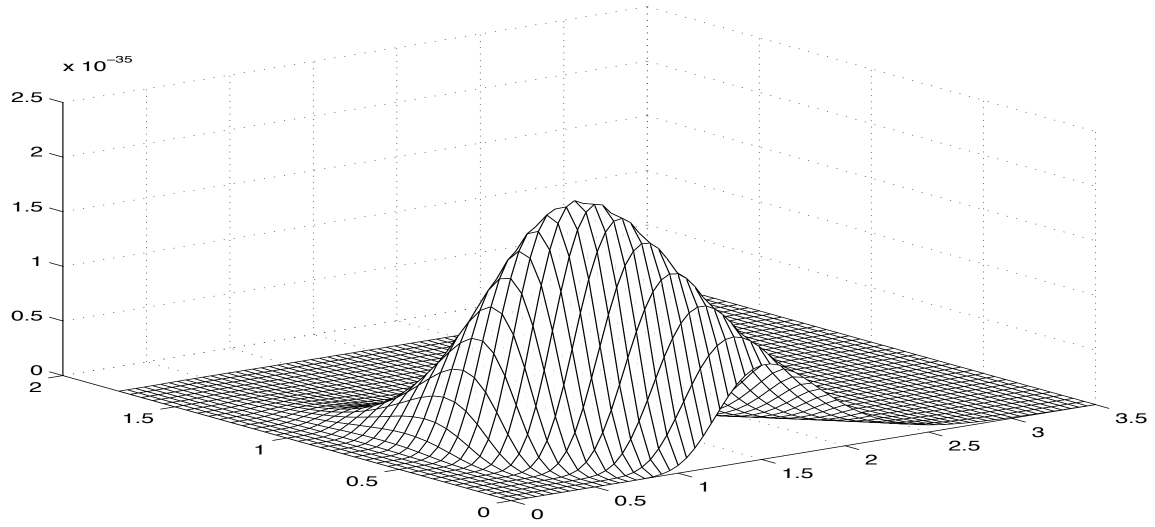

Figure 10.1: Likelihood values for different values of Q and R for the random walk model (values smaller than -90 are truncated). The calculations are based on the first 50 observations of the ones shown in Figure 4.1.

In situations where no simple analytical expressions are available, the estimation procedure becomes more complex. This we will see in the next example, which again consider the random walk model:

Example 3 (random walk, cont.)
Assume now

$$\begin{aligned} x(k) &= x(k-1) + w(k-1) \\ z(k) &= x(k) + v(k). \end{aligned}$$

We will again consider the case $Q(k-1) = Q$ and $R(k) = R$ for all k. Although in principle possible, finding the maximum likelihood estimates analytically will in practice be almost impossible, since an analytical expression of the (log-)likelihood becomes rather complex. For a given set of parameters (Q, R), the (log-)likelihood is however easy to calculate using (10.8). For the simulated data-set shown in Figure 4.1, the log-likelihood was calculated for values of Q in the range $[0.4, 3.0]$ and in the range $[0, 1.25]$ for R. The values are plotted as a perspective plot in Figure 10.1. Note the unimodality of the surface. The likelihood is maximized for $\hat{Q} = 0.85$ and $\hat{R} = 0.35$, which is not far from the true values $Q = 1.0$ and $R = 0.5$. □

10.3 A safe approach: The EM algorithm

A disadvantage of the technique in Section 10.2 is that the numerical maximization of the likelihood function requires expertise in optimization. It may not be that easy to make numerical software work properly, especially if θ is a high-dimensional vector. A safe method which, however, requires more work to implement, is the so-called EM algorithm. This technique is based on the fact that maximum likelihood estimation would have been an easy task numerically if the full likelihood (10.3) was known. Perhaps it works to proceed from an *estimate* of the log-likelihood and determine θ as if the estimate was the true value. In particular, it is possible to replace the unknown log-likelihood (10.5) by some estimate $\hat{l}_{\mathbf{xz}}(\theta)$ where we have replaced all the unknown terms (those involving the system process) with their estimates. We may then maximize $\hat{l}_{\mathbf{xz}}(\theta)$ with respect to θ. The problem is, however, as in the estimation of the system process, that the estimation of $l_{\mathbf{xz}}(\theta)$ itself depends on θ. The situation is thus

$$\theta \qquad \rightarrow \qquad \hat{l}_{\mathbf{xz}}(\theta) \qquad \rightarrow \qquad \text{estimate of } \theta.$$

We may implement this circular relationship recursively. First an initial guess of the parameters $\theta^{(0)}$ produces $\hat{l}_{\mathbf{xz}}(\theta)$ which then revises the parameters to $\theta^{(1)}$ based on maximizing the likelihood above. We may do this again and again, new values of θ leading to new estimates of $\hat{l}_{\mathbf{xz}}(\theta)$ and so forth.

In order to apply the estimation procedure above, some type of estimate for the log-likelihood needs to be constructed. One possibility is to estimate $\mathbf{x}(k)$ by $\hat{\mathbf{x}}(k|N)$, the fixed-interval smoother, and replace the *estimates* in place of $\mathbf{x}(k)$ into the expression (10.5) of $l_{\mathbf{xz}}(\theta)$, giving $\hat{l}_{\mathbf{xz}}(\theta) = l_{\hat{\mathbf{x}}\mathbf{z}}(\theta)$. A problem with such an approach is that the variability in the estimates of the system process is not accounted for. The implication of this is that the estimation procedure will lead to wrong results. In particular, variances involved will be underestimated. The "error" that we made when constructing our estimate for $l_{\mathbf{xz}}(\theta)$ is that we only estimated separate terms in $l_{\mathbf{xz}}(\theta)$ which we plugged in instead of estimating the function itself. A much better approach is to estimate $l_{\mathbf{xz}}(\theta)$ directly by its own expectation:

$$\hat{l}_{\mathbf{xz}}(\theta) = E[l_{\mathbf{xz}}(\theta)|\vec{\mathbf{z}}(N)].$$

For $l_{\mathbf{xz}}(\theta)$, the random components involved are those containing the system process. In general, this expectation can be rather complex. In order to simplify, we will therefore only consider the case $n = m = 1$. Consider first the last term of (10.5).

We have

$$\begin{aligned}
&E[(z(k)-H(k)x(k))^2|\vec{z}(N);\theta=\theta^{(s)}]\\
&\quad=E[(z(k)-H(k)\hat{x}(k|N)+H(k)\hat{x}(k|N)-H(k)x(k))^2|\vec{z}(N);\theta=\theta^{(s)}]\\
&\quad=E[(z(k)-H(k)\hat{x}(k|N))^2|\vec{z}(N);\theta=\theta^{(s)}]\\
&\qquad+E[(H(k)\hat{x}(k|N)-H(k)x(k))^2|\vec{z}(N);\theta=\theta^{(s)}]\\
&\qquad+2E[(z(k)-H(k)\hat{x}(k|N))(H(k)\hat{x}(k|N)-H(k)x(k))|\vec{z}(N);\theta=\theta^{(s)}]\\
&\quad=(z(k)-H(k)\hat{x}(k|N))^2+H(k)^2P(k|N). \qquad (10.11)
\end{aligned}$$

An interesting observation is now that all the unknown quantities in this expression can be calculated by the fixed-interval smoothing algorithm in Section 9.3. Similar arguments lead to

$$E[(x(0)-\mu(0))^2|\vec{z}(N)]=(\hat{x}(0|N)-\mu(0))^2+P(0|N)$$

and

$$\begin{aligned}
&E[(x(k)-\Phi(k-1)x(k-1))^2|\vec{z}(N)]\\
&\quad=(\hat{x}(k|N)-\Phi(k-1)\hat{x}(k-1|N))^2+\\
&\qquad P(k|N)-2\Phi(k-1)\text{cov}[x(k),x(k-1)|\vec{z}(N)]+\Phi(k-1)^2P(k-1|N).
\end{aligned}$$

In this case we need to do some further work to obtain the full expression in that the last term is not directly available. A relative straightforward way of computing this term is however to augment the state variable by the variable of lagged values, $x(k)$, that is,

$$\mathbf{x}^*(k)=\begin{pmatrix} x(k-1)\\ x(k)\end{pmatrix}.$$

Using the original state space model, the augmented model becomes

$$\begin{aligned}
\mathbf{x}^*(k)&=\begin{pmatrix} 0 & 1\\ 0 & \Phi(k-1)\end{pmatrix}\mathbf{x}^*(k-1)+\begin{pmatrix} 0\\ w(k-1)\end{pmatrix},\\
z(k)&=\begin{pmatrix} 0 & 1\end{pmatrix}\mathbf{x}^*(k)+v(k).
\end{aligned}$$

Denote the estimates and covariance matrices obtained by running the fixed-interval smoother on this model by $\hat{\mathbf{x}}^*(k|N)$ and $\mathbf{P}^*(k|N)$. By running the smoothing filter on this augmented state model with $\theta=\theta^{(s)}$, the term $E[(x(k)-\hat{x}(k|N))(x(k-1)-\hat{x}(k-1|N))|\vec{z}(N);\theta=\theta^{(s)}]$ is obtained as the off-diagonal element of the covariance matrix $\mathbf{P}^*(k|N)$. The *estimated* likelihood in the univariate case then becomes

$$\begin{aligned}\hat{l}_{xz}(\theta) =& C - \frac{1}{2}\log|P(0|0)| - \frac{(\hat{x}(0|N)-\mu(0))^2}{2P(0|0)} \\ & - \frac{1}{2}\sum_{k=1}^{N}\log|Q(k-1)| - \sum_{k=1}^{N}\frac{(\hat{x}(k|N)-\Phi(k-1)\hat{x}(k-1|N))^2}{2Q(k-1)} \\ & - \sum_{k=1}^{N}\frac{P(k|N)-2\Phi(k-1)\mathrm{cov}[x(k),x(k-1)|\vec{z}(N)]+\Phi(k-1)^2P(k-1|N)}{2Q(k-1)} \\ & - \frac{1}{2}\sum_{k=1}^{N}\log|R(k)| - \sum_{k=1}^{N}\frac{(z(k)-H(k)\hat{x}(k|N))^2}{2R(k)} \\ & - \sum_{k=1}^{N}\frac{H(k)^2P(k|N)}{2R(k)}. \end{aligned} \tag{10.12}$$

Maximizing this *estimated* log-likelihood, it can be proved, as part of a more general argument, that the process stabilizes when it has been repeated long enough and that $\theta^{(s)}$ then has reached a local maximum on the *true likelihood surface* $L_{\mathbf{z}}(\theta)$. We take this local optimum as our estimate $\hat{\theta}$. The process may be repeated by starting from different initial vectors for θ, and the one ending with the highest value for $L_{\mathbf{z}}(\theta)$ is selected if different $\hat{\theta}$'s are reached in the end. In the example below such multiple solutions did not occur. Finding a global optimum of surfaces with many local optima is a very difficult problem in optimization theory in general.

Example 3 (cont.)
We assume $P(0|0) = \infty$. In this case the formula (10.12) simplifies to

$$\begin{aligned}\hat{l}_{xz}(\theta) =& C - \frac{N}{2}\log(Q) - \sum_{k=1}^{N}\frac{1}{2Q}(\hat{x}(k|N)-\hat{x}(k-1|N))^2 \\ & - \frac{1}{2Q}\sum_{k=1}^{N}\{P(k|N)+P(k-1|N)-2\mathrm{cov}[x(k),x(k-1)|\vec{z}(N)]\} \\ & - \frac{N}{2}\log|R| - \frac{1}{2R}\sum_{k=1}^{N}(z(k)-\hat{x}(k|N))^2 - \frac{1}{2R}\sum_{k=1}^{N}P(k|N)\end{aligned}$$

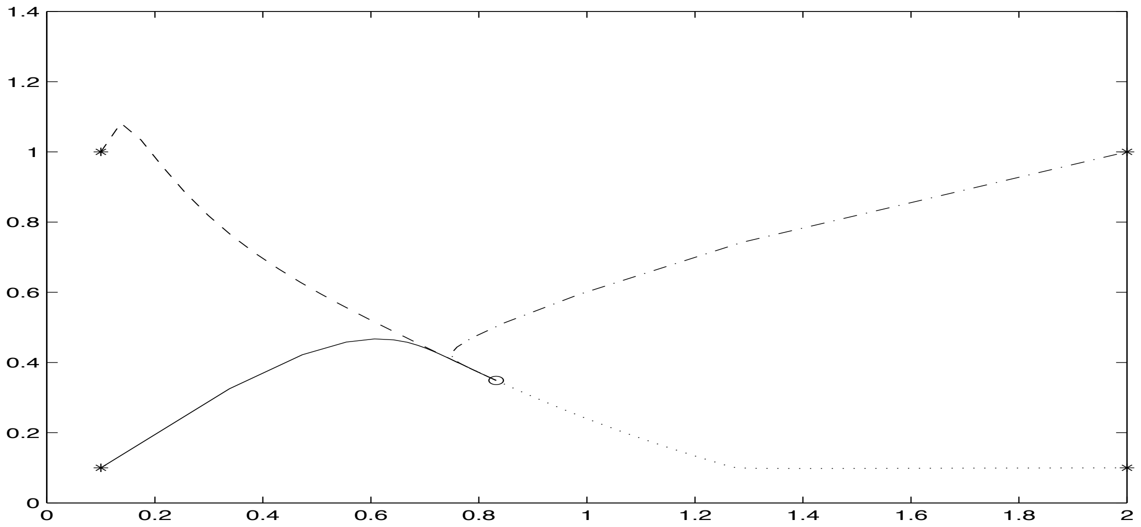

Figure 10.2: Estimates of Q and R using the EM algorithm for different initial values. The four starting values are marked as stars, while the evolution of the parameter estimates as functions of iterations are shown as curves with different types. The common convergence point is marked as a circle.

which is maximized by

$$\begin{aligned}\widehat{Q} =& \frac{1}{N}\sum_{k=1}^{N}(\hat{x}(k|N) - \hat{x}(k-1|N))^2\\ &+ \frac{1}{N}\sum_{k=1}^{N}\{P(k|N) + P(k-1|N) - 2\text{cov}[x(k), x(k-1)|\vec{z}(N)]\},\\ \widehat{R} =& \frac{1}{N}\sum_{k=1}^{N}\{(z(k) - \hat{x}(k|N))^2 + P(k|N)\}.\end{aligned}$$

Comparing these estimates with eqns. (10.6) and (10.7), we see that some additional variance and covariance terms have appeared. This is due to the uncertainty involved in estimating the system process for a given set of parameters.

Using this procedure, we may try to obtain maximum likelihood estimates for Q and R based on the simulated data-set shown in Figure 4.1. Figure 10.2 shows how the estimates changes from iteration to iteration for four different initial values on (Q, R). Note that the estimates in each case converge to the same value $(0.832, 0.348)$. □

In the example above, the likelihood function could be separated into two parts, one involving parameters related to the system process and the other involving parameters related to the observation process. Such a separation is usually obtained also in more complex models, making the maximization part of the algorithm simplify significantly.

10.4 Problems

Exercise 10.1 (Linear regression)
Consider the regression model

$$z(k) = ax(k) + b + v(k),$$

where the random variables $v(k)$ are Gaussian distributed with expectation 0 and standard error σ. Let $x(k) = k$ for $k = 1, \ldots, N$.

Because the x's are known in this case, the likelihood function for the observations $z(1), ..., z(N)$ becomes

$$L(a, b, \sigma^2) = \prod_{k=1}^{N} \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{1}{2\sigma^2}(z(k)-a-bx(k))^2}.$$

(*a*) Why do we not include a distribution for the x's in the likelihood function in this case?

(*b*) Show that the maximum likelihood estimates for a, b and σ^2 are

$$\hat{b} = \frac{\sum_{k=1}^{N}(x(k) - \bar{x})(z(k) - \bar{z})}{\sum_{k=1}^{N}(x(k) - \bar{x})^2}$$
$$\hat{a} = \bar{z} - \hat{b}\bar{x}$$
$$\hat{\sigma}^2 = \frac{1}{N}\sum_{k=1}^{N}(z(k) - \hat{a} - \hat{b}x(k))^2$$

where $\bar{x} = \frac{1}{N}\sum_{k=1}^{N} x(k)$ and similarly for $\bar{z}$.

Specify now $N = 10, a = 2, b = 1$ and $\sigma = 1.5$.

(*c*) Generate in *Matlab* a set of 10 pair of numbers $(x(k), z(k))$ by the routine `randn` — which give *independent* realizations (we hope) — and estimate a, b and σ by the estimates above.

(*d*) Repeat the trial in (*c*) a large number of times and verify that our estimators for a and b are unbiased, that is $E[\hat{a}] = a$ and $E[\hat{b}] = b$.

(*e*) Estimate the variances for $\hat{a}$ and $\hat{b}$. Check these estimates with the theoretical values given by

$$\text{var}[\hat{a}] = \sigma^2 / \sum_{i=1}^{n}(x_i - \bar{x})^2,$$
$$\text{var}[\hat{b}] = \sigma^2 \sum_{i=1}^{n} x_i^2 / (n \sum_{i=1}^{n}(x_i - \bar{x})^2).$$

(f) Make also an analysis of the estimator for σ. Is $\hat{\sigma} = \sqrt{\hat{\sigma}^2}$ unbiased? What about $\hat{\sigma}^2$ itself?

(g) If you wonder how critical the condition of independence is, you may experiment with situations where this is not the case. The number of possibilities is enormous.

Exercise 10.2 (Estimation in AR models)
Consider the model in Exercise 7.2. The maximum likelihood estimators for c and d can in this case be shown to be the solution of a linear equation system of order 2.

(a) Derive the linear equation system.

(Hint: Remember the factorization (10.9) and note that all the conditional distributions are given directly from the original model.)

(b) Generate some realizations of $z(k)$ for $c = 0.8$, $d = 0.15$ and $k = 0, \dots, 100$. Estimate c and d based on these realizations. Examine the results to try to give an indication on if the estimators are biased. Also make some measure one variability of the estimates.

Exercise 10.3 (Maximum likelihood estimation and the EM algorithm)
Consider an extension of the (time-invariant) random walk Example 3 by assuming the system equation is given by

$$x(k) = ax(k-1) + w(k-1)$$

while the observation equation remains unchanged. We are interested in estimating $\theta = (a, Q, R)$.

(a) Assume first that both the x's and the z's are observed. Calculate analytically the maximum likelihood equations for a, Q and R.

(b) Simulate $N = 50$ observations from the model and try out the estimators you obtained in (a).

(c) Use (10.8) to calculate (using *Matlab*) the log-likelihood over a grid of points (a, Q, R) and find the set of values which maximizes the log-likelihood

(d) Find an expression for $\hat{l}(\theta)$ for our extended model.

(e) Implement and run the EM algorithm on your simulated data.

(f) Try different values of N. How do the estimators behave as N increases?

Chapter 11

Non-linear state estimation

Although the theme of this course is *linear* dynamical models, we will include an introduction to non-linear estimation. The theoretical solution to such problems is well known, but calculation of these optimal solutions are in general complicated to perform. Therefore, until recently only rather approximate and *ad hoc* techniques have been available in practice. These standard approaches are based on linearization of the non-linearities involved around some trajectory (possibly a point) in the state space. This leads to the linearized and extended Kalman filters (LKF and EKF, respectively) whose central parts are filters based on linear models that estimate *deviations* from the chosen trajectory. These filters will work well only if the linearization is reasonably accurate. In engineering applications, this may well be the case since the filters then often are parts of a larger feedback loop whose aim is precisely to keep the system at a chosen nominal trajectory.

However, in many cases linearization may not be appropriate but is still used in the absence of any alternative. This situation may now be changing. Over the last years, so-called stochastic simulation techniques from computational statistics, originally developed for quite different purposes, have been used to design practical filters for general non-linear systems. An outline of this emerging field is given here after an introduction to the linearization methods. Finally, some examples are presented where the EKF is compared to solutions obtained from stochastic simulation.

We note in passing that combined state and parameter estimation may also be formulated as a non-linear estimation problem by including the unknown parameters in an augmented state vector. Using this approach, parameter and state estimates are obtained on-line. The likelihood technique in Chapter 10 was restricted to off-line processing. We will consider an example of this approach at the end of the chapter.

11.1 Problem specification

Assume the following non-linear state space model:

$$\mathbf{x}(k) = \mathbf{f}_k(\mathbf{x}(k-1)) + \mathbf{w}(k-1), \qquad \text{(system).} \qquad (11.1a)$$

$$\mathbf{z}(k) = \mathbf{h}_k(\mathbf{x}(k)) + \mathbf{v}(k), \qquad \text{(observations).} \qquad (11.1b)$$

Note that both the system and measurement noise processes are assumed additive. As in the linear case, they will be taken to be white and have zero mean.

The general estimation theory discussed in Chapter 5 is still valid in this case, making the optimal estimate for $\mathbf{x}(k)$ based on $\vec{\mathbf{z}}(k)$, under the minimum mean square error criterion, the conditional expectation:

$$\hat{\mathbf{x}}(k|k) = E[\mathbf{x}(k)|\vec{\mathbf{z}}(k)]. \qquad (11.2)$$

In addition, the covariance matrix for the estimation error is similarly still a valid measure of the uncertainty involved.

In the case of linear models with Gaussian noise terms, the conditional distribution $p(\mathbf{x}(k)|\vec{\mathbf{z}}(k))$ is also Gaussian and fully specified by its means and covariance matrix. In the non-linear situation, several problems do however occur. First, calculation of the conditional expectation is far from simple. Second, for symmetric distributions, such as the Gaussian one, the minimum square error criterion, resulting in that the conditional expectation is the optimal estimate, may be a reasonable one. For more complex distributions, in particular distributions with several modes (peaks), this criterion may give estimates which give quite wrong pictures about the true state. Consider an example where $p(\mathbf{x}(k)|\vec{\mathbf{z}}(k))$ is a mixture of two Gaussian distributions, one centered at -5.0 and the other centered at 5.0. The mixture is displayed in Figure 11.1. The mixture distribution indicates that the true state either is around -5.0 or around 5.0. The mean in the distribution is however 0.0, a very unlikely value!

Our focus on only calculating the mean and covariance matrix of $p(\mathbf{x}(k)|\vec{\mathbf{z}}(k))$ is therefore no longer a viable approach if we want to go beyond the linear situation, and other solutions have to be considered. One solution is to approximate our non-linear model with a linear one and use our algorithms for linear models to compute estimates. This approach has been extensively used for many years, and we will present the main idea in Section 11.2.

Linearization may however give poor results in many cases. A better approach is to focus on the problem of representing and propagating the conditional density itself. The recursion for this density is well known and simple to derive: Start with $\mathbf{x}(0) \sim p(\mathbf{x}(0))$. Then for $k = 1, 2, ...$, we have

$$p(\mathbf{x}(k)|\vec{\mathbf{z}}(k-1)) = \int_{-\infty}^{\infty} p(\mathbf{x}(k)|\mathbf{x}(k-1))p(\mathbf{x}(k-1)|\vec{\mathbf{z}}(k-1))\, d\mathbf{x}(k-1), \qquad (11.3)$$

$$p(\mathbf{x}(k)|\vec{\mathbf{z}}(k)) = C_k p(\mathbf{z}(k)|\mathbf{x}(k))p(\mathbf{x}(k)|\vec{\mathbf{z}}(k-1)), \qquad (11.4)$$

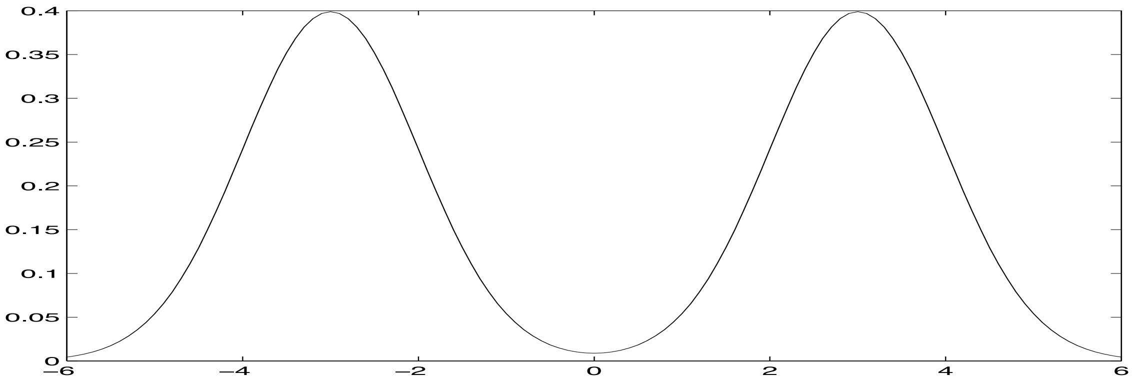

Figure 11.1: Example of a distribution which is a mixture of two Gaussian ones, each with standard deviation equal to 1.0 but with expectations ±5.0.

where $1/C_k = p(\mathbf{z}(k)|\vec{\mathbf{z}}(k-1))$ is a normalization constant once $\vec{\mathbf{z}}(k)$ is known.

Eq. (11.3) readily follows from the fact that

$$p(\mathbf{x}(k)|\mathbf{x}(k-1)) = p(\mathbf{x}(k)|\mathbf{x}(k-1), \vec{\mathbf{z}}(k-1)).$$

To see eq. (11.4), note that $p(\mathbf{x}(k)|\vec{\mathbf{z}}(k)) = p(\mathbf{x}(k), \mathbf{z}(k)|\vec{\mathbf{z}}(k-1))C_k$ and apply Bayes rule once more to the numerator.

Attacking these equations directly has turned out to be an intractable problem at least when the dimension of the state space is not one or two. In higher dimensions, numerical integration gets very time consuming even on present day computers. The computational burden has a tendency to grow exponentially with the dimension, thus rendering the direct approach impractical. In such cases, one often speaks of the *curse of dimensionality.*

11.2 The linearized and extended Kalman filters

The idea behind the linearized Kalman filter is that the system stays reasonably close to some desired, nominal trajectory $\mathbf{x}^N(k)$ given by the deterministic equation $\mathbf{x}^N(k) = \mathbf{f}_k(\mathbf{x}^N(k-1))$. Note that this is not the same as the expected trajectory $\{E[\mathbf{x}(k)]\}$ since in general $E[\mathbf{f}_k(\mathbf{x}(k))] \neq \mathbf{f}_k(E[\mathbf{x}(k)])$. Think of the nominal trajectory as a given first order approximation to what the system is doing. For example, in the moving body case this would correspond to the situation where we knew *approximately* the position and velocity at any time. The point is then to try to estimate *deviations* from the nominal trajectory using a Kalman filter.

Define the *state vector deviation* as $\delta\mathbf{x}(k) = \mathbf{x}(k) - \mathbf{x}^N(k)$ and the *measurement deviation* as $\delta\mathbf{z}(k) = \mathbf{z}(k) - \mathbf{h}_k(\mathbf{x}^N(k))$. Assuming that $\mathbf{f}_k$ can be expanded in a

Taylor series, we have

$$\begin{aligned}\mathbf{x}(k) =&\mathbf{f}_k(\mathbf{x}(k-1)) + \mathbf{w}(k-1)\\ =&\mathbf{f}_k(\mathbf{x}^N(k-1)) + \frac{\partial \mathbf{f}_k(\mathbf{x})}{\partial \mathbf{x}}_{|\mathbf{x}=\mathbf{x}^N(k-1)} \delta\mathbf{x}(k-1) + \mathbf{w}(k-1) + h.o.t.\end{aligned}$$

where *h.o.t.* are higher order terms in the Taylor expansion. This leads to the approximate linear equation in the “state” $\delta\mathbf{x}$

$$\delta\mathbf{x}(k) \approx \mathbf{\Phi}(k-1)\delta\mathbf{x}(k-1) + \mathbf{w}(k-1), \tag{11.5}$$

where

$$\mathbf{\Phi}(k-1) = \frac{\partial \mathbf{f}_k(\mathbf{x})}{\partial \mathbf{x}}_{|\mathbf{x}=\mathbf{x}^N(k-1)}. \tag{11.6}$$

To derive the corresponding measurement equation, we proceed in a similar way:

$$\begin{aligned}\mathbf{z}(k) =&\mathbf{h}_k(\mathbf{x}(k)) + \mathbf{v}(k)\\ =&\mathbf{h}_k(\mathbf{x}^N(k)) + \frac{\partial \mathbf{h}_k(\mathbf{x})}{\partial \mathbf{x}}_{|\mathbf{x}=\mathbf{x}^N(k)} \delta\mathbf{x}(k) + \mathbf{v}(k) + h.o.t.\end{aligned}$$

leading to

$$\delta\mathbf{z}(k) \approx \mathbf{H}(k)\delta\mathbf{x}(k) + \mathbf{v}(k), \tag{11.7}$$

where

$$\mathbf{H}(k) = \frac{\partial \mathbf{h}_k(\mathbf{x})}{\partial \mathbf{x}}_{|\mathbf{x}=\mathbf{x}^N(k)}. \tag{11.8}$$

Equations (11.5) and (11.7) comprise the state space model for deviations from the nominal trajectory and a standard Kalman filter can be used to compute $\delta\hat{\mathbf{x}}(k|k)$. The estimate of the actual state vector is obtained as

$$\hat{\mathbf{x}}(k|k) = \delta\hat{\mathbf{x}}(k|k) + \mathbf{x}^N(k). \tag{11.9}$$

The variance of the errors in this estimate is provided by the error covariance matrix for $\delta\hat{\mathbf{x}}(k|k)$. Of course, these variances are only meaningful as long as the linearization is valid. The higher order terms in the Taylor expansions may be regarded as a kind of modeling errors that may or may not lead to divergence of the linearized filter estimates (see Section 8.1 for an example of modeling errors causing divergence). As time increases, one would expect the actual trajectory, driven by the system noise, to deviate more from the nominal one and the linearization to become less valid.

In the extended Kalman filter (EKF), one tries to alleviate this potential problem by linearizing about the currently *best estimate* of the state vector instead of a fixed nominal trajectory. Computing the state estimate, one also avoids linearizations in the prediction and measurement update steps by using the actual non-linear expressions. The state estimates are thus computed as follows:

$$\hat{\mathbf{x}}(k|k-1) = \mathbf{f}_k(\hat{\mathbf{x}}(k-1|k-1)), \tag{11.10}$$

$$\hat{\mathbf{x}}(k|k) = \hat{\mathbf{x}}(k|k-1) + \mathbf{K}(k)(\mathbf{z}(k) - \mathbf{h}_k(\hat{\mathbf{x}}(k|k-1))). \tag{11.11}$$

The gain matrix is obtained through the standard covariance recursion (eqs. (6.13)) now based on the matrices

$$\mathbf{\Phi}(k-1) = \frac{\partial \mathbf{f}_k(\mathbf{x})}{\partial \mathbf{x}}_{|\mathbf{x}=\hat{\mathbf{x}}(k-1|k-1)} \tag{11.12}$$

and

$$\mathbf{H}(k) = \frac{\partial \mathbf{h}_k(\mathbf{x})}{\partial \mathbf{x}}_{|\mathbf{x}=\hat{\mathbf{x}}(k|k-1)}. \tag{11.13}$$

The EKF may be further refined by recomputing $\mathbf{H}(k)$ through linearization around $\hat{\mathbf{x}}(k|k)$, then derive a new gain matrix and repeat the calculation in eq. (11.11) substituting $\mathbf{x}(k|k)$ on the right hand side. Clearly, this can be repeated as many times as desired. The result is the *iterated* extended Kalman filter. However, we will not go into these potential improvements as they are all more or less *ad hoc*. Extensive computer simulations are necessary in each application to "tune" the linearization techniques.

11.3 Monte Carlo filtering

Linearization, as described in the previous section, may give poor results. This has however been the only tool available from the inception of the Kalman filter in 1960 to the early 1990's when **Monte Carlo filtering** appeared. As noted in Section 11.1, numerical integration is not the solution because of the computational burden. The ideas behind the Monte Carlo filter are different in that one does not try to compute the conditional distributions themselves. Instead, the distributions are *estimated* by simulated *sample* sets which are updated recursively for each time step. From the samples, one may estimate any number of parameters for the distributions (including their mean and covariance matrices). Note that this approach does not require the system and measurement noise in eq. (11.1) to be additive. As it turns out, the system noise can enter in quite general ways as can the measurement noise as long as the likelihood $p(\mathbf{z}(k)|\mathbf{x}(k))$ can be computed during the measurement update step.

The question is how to sample. Suppose samples of $\mathbf{x}(0)$ have been generated (by drawing from $p(\mathbf{x}(0))$). By passing these through the system equation, combining them with samples of the system noise $\mathbf{w}(0)$ as required, samples from $p(\mathbf{x}(1))$ are generated. This corresponds to the *prediction* step in Kalman filtering. There is an additional *update* step, where the measurement $\mathbf{z}(1)$ is used to transform the samples from $p(\mathbf{x}(1))$ into samples from $p(\mathbf{x}(1)|\vec{\mathbf{z}}(1))$.

In general, the Monte Carlo filtering problem is how to convert a collection of samples from $p(\mathbf{x}(k-1)|\vec{\mathbf{z}}(k-1))$ into one from $p(\mathbf{x}(k)|\vec{\mathbf{z}}(k))$. The crucial issue is the update step and we will only consider one method - so-called **rejection sampling** - for carrying this out. Another method, **sampling/importance resampling**, has also been applied but it seems at present less promising. The update step introduces a requirement that motivates the way the prediction step is carried out. Update is therefore discussed first.

11.3.1 The update step

Suppose it is possible to sample from the prediction density $p(\mathbf{x}(k)|\vec{\mathbf{z}}(k-1))$. Rejection sampling is a general technique from sampling theory that allows one to integrate the effect of the observation $\mathbf{z}(k)$ thus obtaining a sample from $p(\mathbf{x}(k)|\vec{\mathbf{z}}(k))$. Assume knowledge of a constant M such that

$$p(\mathbf{z}(k)|\mathbf{x}(k)) \leq M, \qquad \text{for all } \mathbf{x}(k). \tag{11.14}$$

The conditional likelihood $p(\mathbf{z}(k)|\mathbf{x}(k))$ is typically easy to compute, at least up to a constant of proportionality (which is enough). The following algorithm returns a random vector from $p(\mathbf{x}(k)|\vec{\mathbf{z}}(k))$.

Rejection sampling (RS):

```
 Repeat
     sample $\mathbf{x}^*(k)$ from $p(\mathbf{x}(k)|\vec{\mathbf{z}}(k-1))$,
     sample $U^*$ from the uniform $(0,1)$,
 until $U^* \leq p(\mathbf{z}(k)|\mathbf{x}^*(k))/M$.
 Return $\mathbf{x}^*(k)$.
```

The stopping rule ensures that $\mathbf{x}^*(k)$ on return has the right distribution $p(\mathbf{x}(k)|\vec{\mathbf{z}}(k))$ although it originally came from a different one. When the algorithm is run B times, an independent sample set of size B is obtained. Note that the normalization constant C_k in (11.4) is not needed.

If the number of tries before a sample is accepted becomes too large, the algorithm will become inefficient. An upper limit on the number of tries cannot be stated in principle, but the expected number of tries is proportional to M. For high efficiency, M in (11.14) should therefore be as small as possible. The average number of tries per sample will be sued as a measure of performance.

Proof: To prove the algorithm, we need to show that the output $\mathbf{x}^*(k)$ has a probability distribution equal to $p(\mathbf{x}(k)|\vec{\mathbf{z}}(k))$. Note however that since

$$p(\mathbf{x}(k)|\vec{\mathbf{z}}(k)) \propto p(\mathbf{x}(k)|\vec{\mathbf{z}}(k-1))p(\mathbf{z}(k)|\mathbf{x}(k)) \tag{11.15}$$

and a probability distribution always has to integrate to one, it is enough to show that the distribution of $\mathbf{x}^*(k)$ is proportional to the right hand side of (11.15). Note that we have to condition on that $\mathbf{x}^*(k)$ is accepted. From standard probability theory, we have

$$p(\mathbf{x}^*(k)|\mathbf{x}^*(k) \text{ accepted}) = \frac{\partial}{\partial \mathbf{x}} \Pr(\mathbf{x}^*(k) \leq \mathbf{x}|\mathbf{x}^*(k) \text{ accepted}).$$

Now, from the way $\mathbf{x}^*(k)$ is generated, we get

$$\begin{aligned}&\Pr(\mathbf{x}^*(k) \leq \mathbf{x} \cap \mathbf{x}^*(k) \text{ accepted})\\ &\qquad = \frac{1}{M}\int_{\mathbf{y}\leq\mathbf{x}} p(\mathbf{x}(k)=\mathbf{y}|\vec{\mathbf{z}}(k-1))p(\mathbf{z}(k)|\mathbf{x}(k)=\mathbf{y})\, d\mathbf{y}.\end{aligned}$$

Therefore

$$\Pr(\mathbf{x}^*(k) \text{ accepted}) = \frac{1}{M}\int_{\mathbf{y}\in\mathcal{R}^n} p(\mathbf{x}(k)=\mathbf{y}|\vec{\mathbf{z}}(k-1))p(\mathbf{z}(k)|\mathbf{x}(k)=\mathbf{y})\, d\mathbf{y}.$$

Using Bayes rule, we have

$$\begin{aligned}&\Pr(\mathbf{x}^*(k) \leq \mathbf{x}|\mathbf{x}^*(k) \text{ accepted})\\ &\qquad =\frac{\int_{\mathbf{y}\leq\mathbf{x}} p(\mathbf{x}(k)=\mathbf{y}|\vec{\mathbf{z}}(k-1))p(\mathbf{z}(k)|\mathbf{x}(k)=\mathbf{y})\, d\mathbf{y}}{\int_{\mathbf{y}\in\mathcal{R}^n} p(\mathbf{x}(k)=\mathbf{y}|\vec{\mathbf{z}}(k-1))p(\mathbf{z}(k)|\mathbf{x}(k)=\mathbf{y})\, d\mathbf{y}}\\ &\qquad =C\int_{\mathbf{y}\leq\mathbf{x}} p(\mathbf{x}(k)=\mathbf{y}|\vec{\mathbf{z}}(k-1))p(\mathbf{z}(k)|\mathbf{x}(k)=\mathbf{y})\, d\mathbf{y}\end{aligned}$$

The scaling factor M disappears and by differentiating the right hand side, the result shows that $\mathbf{x}^*(k)$ has a density proportional to $p(\mathbf{x}(k)|\vec{\mathbf{z}}(k-1))p(\mathbf{z}(k)|\mathbf{x}(k))$ as desired.

The key to a successful implementation of the algorithm is that the samples $\mathbf{x}^*(k)$ from $p(\mathbf{x}(k)|\vec{\mathbf{z}}(k-1))$ do not generally produce insignificant values of the likelihood $p(\mathbf{z}(k)|\mathbf{x}^*(k))$ (relative to M). Thus, for the method to work well, the likelihood should preferably cover much of the prior density. (In rejection sampling terminology, our likelihood would be called the "envelope".) To what degree this happens in a filtering problem depends on the $\mathbf{z}$ actually measured, the function $\mathbf{h}$, and the measurement noise $\mathbf{v}$. The following examples will illustrate this point.

Examples of rejection sampling

To be specific, assume a one dimensional linear system with both $x(0)$ and $w(0)$ being Gaussian, implying that $x(1)$ is also Gaussian, say $x(1) \sim \mathcal{N}(m, q)$. Let us evaluate $p(x(1)|z(1))$ for two different non-linear measurement functions $h_1(x(1))$ while assuming that $v(1)$ is Gaussian with zero mean and variance r. The likelihood is then simply

$$p(z(1)|x(1)) = \frac{1}{\sqrt{2\pi r}} \exp\left\{-\frac{1}{2r}(z(1) - h_1(x(1)))^2\right\} \tag{11.16}$$

whose maximum value is $M = 1/\sqrt{2\pi r}$.

Note first that if r is large compared to the values taken by $(z(1) - h_1(x(1)))^2$, the likelihood will always be close to M and most samples from the prediction density will be accepted. In this case, the measurement carries little information and the updated distribution essentially becomes equal to the predicted.

Throughout all the following experiments, we will assume $m = 2, q = 1$, and $r = 0.2$.

Assume that $x(1)$ is observed indirectly through two observes, one having $h_1(x(1)) = x(1)^2$ and the other $h_1(x(1)) = 1/x(1)^2$. For each measurement function, we first consider the case where $z(1) = 4.0$ for the first observer while $z(1) = 0.25$ for the second one. Note that if no observation error is present, both observations correspond to the case where $x(1) = 2.0$ (the most likely value in the predictive distribution) or $x(1) = -2.0$.

The second case we will consider is when $z(1) = 0.25$ for the first observer while $z(1) = 4.0$ for the second one. This would correspond to $x(1) = \pm 0.5$ (both unlikely values in the predictive distribution) if no observation errors were involved.

In both cases we will illustrate how the predictive distribution $p(x(1))$ is updated when observation $z(1)$ becomes available. In all the figures, the distribution for $x(1)$ is approximated by samples from it. The distributions are represented by $N = 500$ samples in all cases. Because we in this simple situation are able to calculate the *true* distribution, this is added to the plots in order to illustrate the performance of the simulation procedure.

The performance (measured as the number of samples that have to be generated before one is accepted) of the rejection sampling algorithm will depend on the match between the predictive distribution $p(x(1))$ and the distribution for the observation $p(z(1)|x(1))$. The latter is therefore plotted together with the former.

With $z(1) = 4.0$ and $h_1(x(1)) = x(1)^2$, the likelihood in Figure 11.2 has two sharp peaks at 2.0 and -2.0. With the comparatively small measurement noise in this example ($r = 0.2$), the measurement $z(1)$ will be dominated by the $h_1(x(1))$ term but from the likelihood it is impossible to determine whether $x(1)$ is 2.0 or -2.0. However, since practically all the mass of the predicted density lies to the right of -2.0, the updated density is quite narrow, unimodal and centered around 2.0. Note that the likelihood is always symmetric about the origin in this case. For positive $z(k)$ it is bimodal and for negative $z(k)$ it is unimodal. The narrowness of the likelihood caused an average of 9.4 repetitions in the rejection sampling algorithm in order to accept a sample.

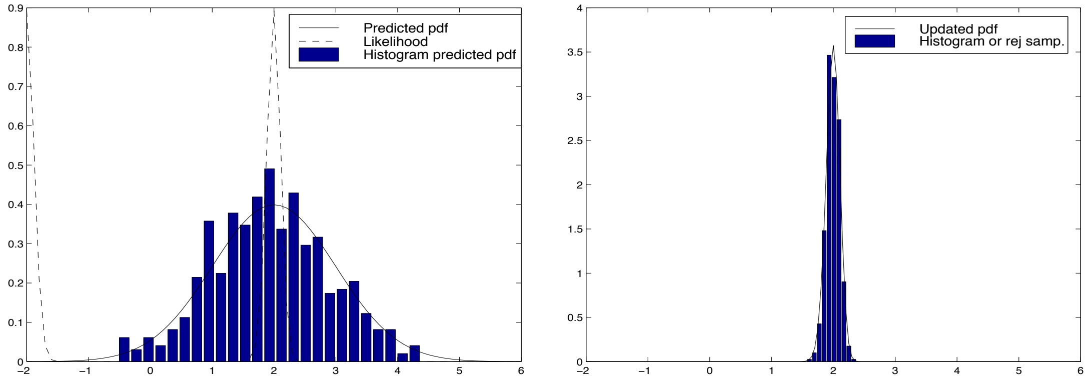

Figure 11.2: Measurement function $h_1(x(1)) = x(1)^2$ with $z(1) = 4.0$. Left: Predicted distribution with histogram based on $B = 500$ samples and likelihood. Right: Updated distribution also with histogram obtained through rejection sampling.

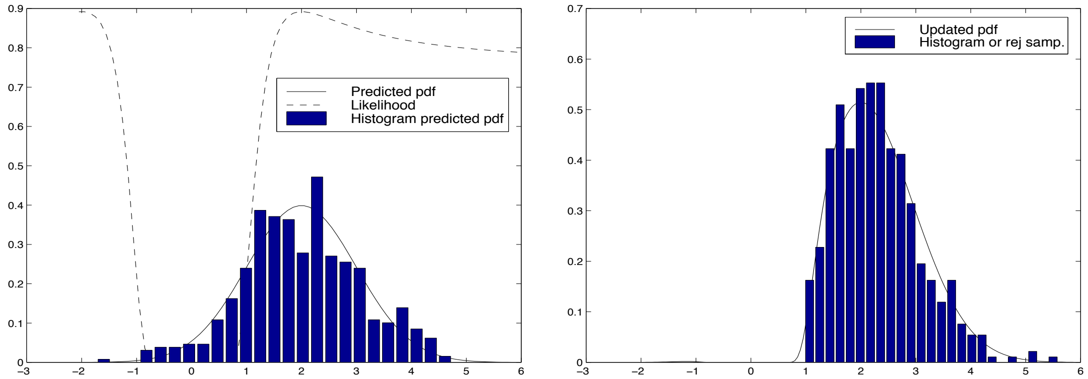

Figure 11.3: Measurement function $h_1(x(1)) = 1/x(1)^2$ with $z(1) = 0.25$. Left: Predicted distribution with histogram based on $B = 500$ samples and likelihood. Right: Updated distribution also with histogram obtained through rejection sampling.

Turning to $h_1(x(1)) = \frac{1}{x^2(1)}$ and $z(1) = 0.25$ in Figure 11.3, this term will not dominate $z(1)$ since the measurement noise has standard deviation $\sqrt{0.2} \approx 0.44$. With this $z(1)$, the likelihood will exclude areas around 0 and attach significant weight to values larger than 1 in absolute value. The result is an updated density considerably wider than in the previous case. Since there is a better match between the predictive distribution and the likelihood in this case, the average number of repetitions in the rejection sampling algorithm now reduced to 1.3.

In Figures 11.4 and 11.5, $z(1) = 0.25$ and $= 4.0$ for the first and second observer, respectively. Compared to the previous case, this produces a smaller measurement for $h_1(x(1)) = (x(1))^2$ and a larger one for $h_1(x(1)) = \frac{1}{x^2(1)}$. This time the latter measurement function produces the sharpest estimate although the updated distribution has a small second peak at -0.5. For the first observed, an average of 8.0 repetitions

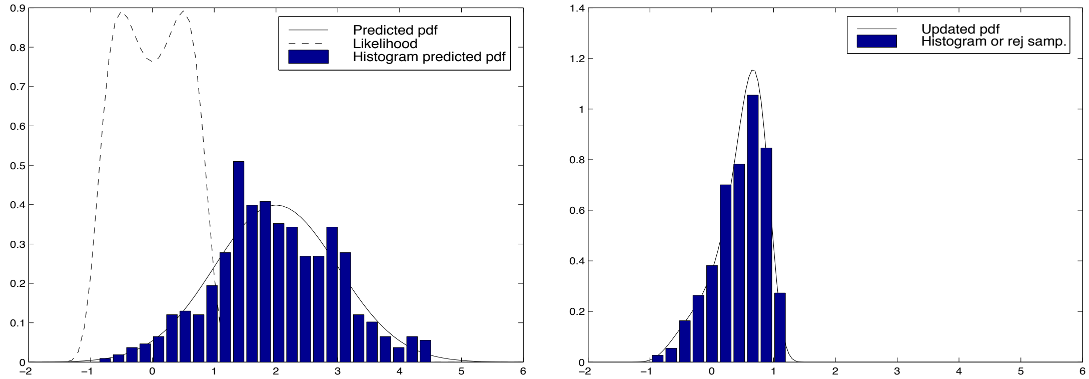

Figure 11.4: Measurement function $h_1(x(1)) = x(1)^2$ with $z(1) = 0.25$. Left: Predicted distribution with histogram based on $B = 500$ samples and likelihood. Right: Updated distribution also with histogram obtained through rejection sampling.

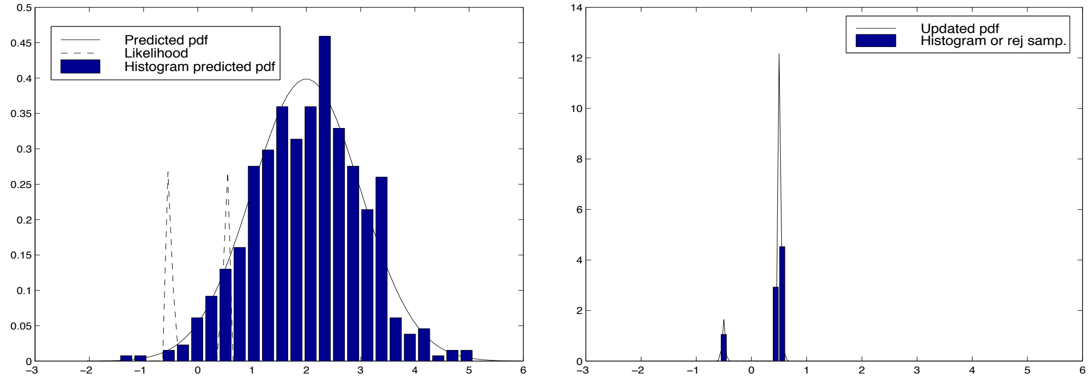

Figure 11.5: Measurement function $h_1(x(1)) = 1/x(1)^2$ with $z(1) = 4.0$. Left: Predicted distribution with histogram based on $B = 500$ samples and likelihood. Right: Updated distribution also with histogram obtained through rejection sampling.

were needed for generating a sample while the corresponding number for the second observer was as high as 100.6! The narrowness of the likelihood is again the reason for this. Note also that we have an increase in the number of repetitions for both observers compared to the first case. This is because the observations in the last case correspond to more unlikely $x(1)$ values in the predictive distribution.

11.3.2 Monte Carlo prediction

The next question is how to sample from the prediction density $p(\mathbf{x}(k)|\vec{\mathbf{z}}(k-1))$ when B realizations $\mathbf{x}^{*b}(k-1)$, $b = 1, \ldots, B$ from the *preceding* update density $p(\mathbf{x}(k-1)|\vec{\mathbf{z}}(k-1))$ are at disposal. All of $\mathbf{x}^{*1}(k-1), \ldots, \mathbf{x}^{*B}(k-1)$ may be inserted into the system equation (11.1a). When system noise is drawn, B samples from $p(\mathbf{x}(k)|\vec{\mathbf{z}}(k-1))$

are generated, but that is not enough. The rejection update algorithm above clearly requires generating samples from $p(\mathbf{x}(k-1)|\vec{\mathbf{z}}(k-1))$ many (at least B, but typically more) times to produce B samples from $p(\mathbf{x}(k)|\vec{\mathbf{z}}(k))$, since many of the samples are rejected. Starting with a very large number B for $k = 0$ and then reducing B with increasing k is not a solution. That would imply unnecessary much computational work in the beginning and we would eventually run out of samples in an unpredictable way.

A rational approach to this problem is to use the B samples to *estimate* $p(\mathbf{x}(k-1)|\vec{\mathbf{z}}(k-1))$ and then draw samples from the estimated distribution as many times as we want. Density estimation is a highly developed area of statistics with a host of possible methods available.

As a first approach, one might believe that sampling could be performed from the histogram representing the updated distribution. This would mean to attach probability $1/B$ to each of the B samples and sample with replacement. However, when there is little system noise, the Monte Carlo filter would propagate almost the same sample values from one time step to the next, and the number of *different* samples would tend to collapse to a few or even a single value.

Consider instead a smoothed version of the histogram. The procedure is simple to carry out. First derive a set of corrected samples each assigned a probability $1/B$:

$$\mathbf{x}_*^{*b}(k-1) = \bar{\mathbf{x}}^{*B}(k-1) + \{(1-\lambda^2)B/(B-1)\}^{1/2}(\mathbf{x}^{*b}(k-1) - \bar{\mathbf{x}}^{*B}(k-1))$$

for $b = 1, 2, \ldots, B$. Here $\bar{\mathbf{x}}^{*B}(k-1)$ is the mean vector of $\{\mathbf{x}^{*b}(k-1)\}$, $\mathbf{P}^{*B}(k-1)$ the corresponding sample covariance matrix, and $\lambda \in [\,0, 1\,]$ is a user selected parameter. It is easy to verify that the distribution has mean $\bar{\mathbf{x}}^{*B}(k-1)$ and covariance matrix $(1-\lambda^2)\mathbf{P}^{*B}(k-1)$.

To generate a sample from the distribution we want, draw a corrected sample $\mathbf{x}_*^{*b}(k-1)$ and add an independent Gaussian variable ϵ_λ^* with mean zero and covariance matrix $\lambda^2\mathbf{P}^{*B}(k-1)$. Clearly, the resulting distribution has a mean and covariance matrix equal to $\bar{\mathbf{x}}_*^{*b}$ and $\mathbf{P}^{*B}(k-1)$, respectively. The distribution is a *continuous* one and thus represents a *smoothed* version of the histogram. Note that there is no restriction for ϵ_λ^* to be Gaussian. Any continuous distribution might do (and will typically not influence much on the results), but we will for simplicity concentrate on the Gaussian one. The choice of λ is more crucial, and has been given much consideration in the literature about density estimation. In particular it must depend on both the uncertainty involved in the samples *and* on the sample size. One possible choice is to use

$$\lambda = CB^{-\frac{1}{n+4}}\sqrt{|\mathbf{P}^{*B}(k-1)|} \tag{11.17}$$

where C is some constant that can be specified by inspection of the results.

In detail, the algorithm runs as follows:

Sampling from the (estimated) prediction density.
Draw b with equal probabilities from $\{1, \ldots, B\}$.
Sample ϵ^*_λ from $\mathcal{N}(\mathbf{0}, \lambda^2 \mathbf{P}^{*B}(k-1))$.
Compute

$$\mathbf{x}^*(k-1) = \bar{\mathbf{x}}^{*B}(k-1) + \left\{ \frac{(1-\lambda^2)B}{B-1} \right\}^{1/2} (\mathbf{x}^{*b}(k-1) - \bar{\mathbf{x}}^{*B}(k-1)) + \epsilon^*_\lambda$$

Sample $\mathbf{w}^*(k-1)$ and return $\mathbf{x}^*(k) = \mathbf{f}(\mathbf{x}^*(k-1)) + \mathbf{w}^*(k-1))$;

The complete Monte Carlo filter is obtained by combining the prediction samples generated in this way with the rejection algorithm in the previous section.

11.4 Examples of non-linear filtering

As noted above, the EKF has been the standard approach for generating approximate solutions to non-linear filtering problems. Except in simple cases, the degree of approximation has been hard or impossible to assess. The Monte Carlo techniques offer the hope of radically better solutions in that the underlying distributions themselves are approximated. Much work remains to be done regarding the properties of these filters. However, as a rule of thumb we may say that sufficient accuracy has been obtained once the results do not change appreciably by increasing B.

Example 5
Example from [5] originally proposed by [10]. Consider the system

$$\begin{aligned} x(k) &= 0.5x(k-1) + \frac{25x(k-1)}{1+x^2(k-1)} + 8\cos(1.2(k-1)) + w(k-1), \\ z(k) &= \frac{x^2(k)}{20} + v(k). \end{aligned}$$

Disregarding the cosine term and the system noise, we have a stable non-linear difference equation. The cosine term causes the system to undergo periodic oscillations and the system noise makes the oscillations random. As noted above, the squared term in the measurement equation causes the likelihood to be bimodal for positive z and unimodal for negative z. The example is designed so that z is positive most of the time. This will be one reason why the EKF is not expected to work very well. The EKF may be thought of as approximating the predicted and updated distributions by Gaussians and we know that with the quadratic measurement equation this may be inappropriate. In the example, $q = 10, r = 1$ and $x(0) \sim N(0, 5)$. The number of updated samples is set at $B = 1000$ (which is too much really).

Figure 11.6 compares the results of the EKF and the Monte Carlo filter. Clearly, the latter is superior. The EKF generally exhibits fairly large deviations from the

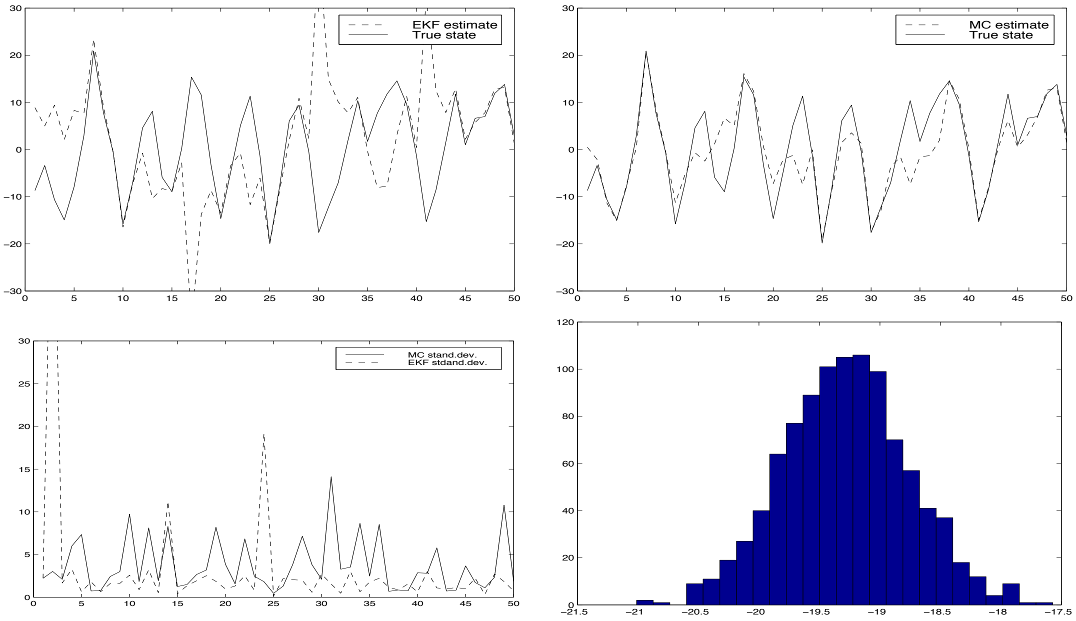

Figure 11.6: Example from Gordon et al. (1993). Upper left panel: State estimate from EKF and true state. Upper right panel: State estimate from MC filter with true state and measurements. Lower left panel: Standard deviation of the EKF estimate error as computed by the EKF and the corresponding standard deviation from the MC filter. Lower right panel: Histogram of samples from $p(x(25)|\vec{\mathbf{z}}(25))$.

true state with some extreme estimates at $k = 17, 30$ and 41. Note also that the estimate of the EKF standard deviation does not reflect the real error; the EKF is over-optimistic regarding its own performance. The Monte Carlo filter, on the other hand, follows the true state quite well and is realistic regarding its performance. Note that the estimates in most cases have the right signs.

In the Monte Carlo filter, we have used the mean of samples from $p(x(k)|\vec{\mathbf{z}}(k))$ as an estimate of $x(k)$. As noted above, the mean correspond to the optimal estimator under the minimum square error criterion. Such a criterion is however only sensible if the distribution is unimodal and fairly symmetric. The lower right panel of Figure 11.6 shows the histogram of samples from $p(x(25)|\vec{z}(25))$. The distribution *does* look well-behaved in this case, implying that the mean might be a reasonable estimate. □

Example 6 (Direction measurements)
The next example (from [4]) is two-dimensional with a linear system equation and a non-linear measurement equation. The example models an observer, rotating on the unit circle with constant angular velocity, measuring the *direction* to an object moving randomly in the plane. The aim is to estimate the position $(x_1, x_2)^T$ of the

object based only on the directional information. The model is given by

$$\begin{aligned}x_1(k) &= 0.5x_1(k-1) + w_1(k-1)\\ x_2(k) &= x_2(k-1) + w_2(k-1)\end{aligned}$$

and

$$z(k) = \text{atan}\left(\frac{x_2(k) - \sin(k)}{x_1(k) - \cos(k)}\right) + v(k)$$

Here the covariance matrices for the noise terms are given by

$$\mathbf{Q} = \left(\begin{array}{cc} 0.1 & 0.05 \\ 0.05 & 0.1 \end{array}\right) \quad \text{and} \quad R = 0.1.$$

Note that the system equation is linear in this case, making

$$\mathbf{\Phi}(k-1) = \left(\begin{array}{cc} 0.5 & 0 \\ 0 & 1 \end{array}\right),$$

that is constant for all k in the EKF. Further,

$$\begin{aligned}\mathbf{H}(k) =& \frac{\partial}{\partial \mathbf{x}} \mathbf{h}_k(\mathbf{x})_{|\mathbf{x}=\hat{\mathbf{x}}(k|k-1)}\\ =& C(\hat{\mathbf{x}}(k|k-1)) \left(\begin{array}{cc} -\hat{x}_2(k|k-1) + \sin(k), & \hat{x}_1(k|k-1) - \cos(k) \end{array}\right)\end{aligned}$$

where

$$C(\hat{\mathbf{x}}(k|k-1)) = \frac{1}{(\hat{x}_1(k|k-1) - \cos(k))^2 + (\hat{x}_2(k|k-1) - \sin(k))^2}$$

The upper panel of Figure 11.7 shows the results obtained by EKF for data simulated from the model. In particular for $x_2(k)$ (but also for $x_1(k)$) the result is poor. The estimate *diverge* although the true system process is stable! The Monte Carlo filter was run with $B = 1000$, and for each k, the mean of the simulated samples from $p(\mathbf{x}(k)|\mathbf{z}(k))$ was used as an estimate for $\mathbf{x}(k)$. These estimates are shown in the middle panel of Figure 11.7. As for the previous example, the Monte Carlo filter obtains much better results, although even the Monte Carlo filter has problems estimating $x_1(k)$ reasonable correct. The lower panel shows the estimated variances involved. Note that the variance of $x_2(k)$ obtained by the MC filter seems to diverge in this case. If this actually can happen is still an open question.

Also in this case we want to inspect the actual distribution of $\mathbf{x}(k)$ in order to evaluate if the mean is a realistic estimate. Figure 11.8 shows the histograms of samples from $p(\mathbf{x}(25)|\vec{z}(25))$. Since both histograms are unimodal and fairly symmetric, it is reasonable to use the mean as an estimate of the state. □

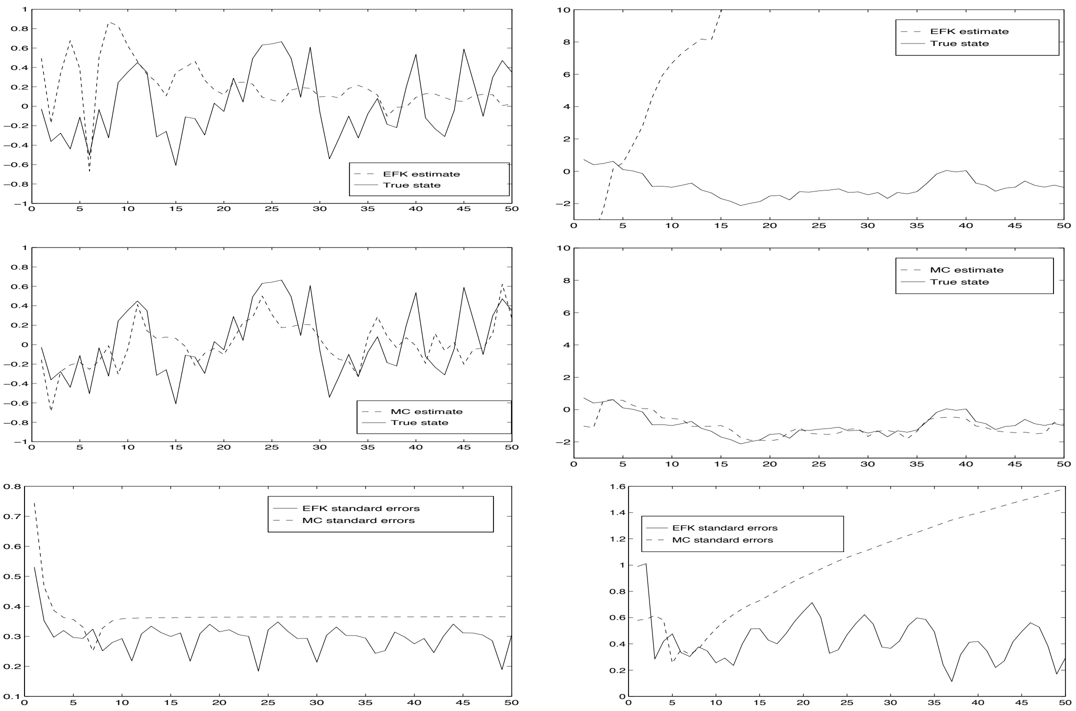

Figure 11.7: Direction measurements. Upper panel: State estimate from EKF and true state. Middle panel: State estimate from MC filter with true state and measurements. Lower panel: Standard deviation of the EKF estimate error as computed by the EKF and the corresponding standard deviation from the MC filter.

Example 7 (Parameter estimation)
The third example is a parameter estimation problem. The original system is linear

$$\begin{aligned} x_1(k) &= ax_1(k-1) + w_1(k-1), \\ z(k) &= x_1(k) + v(k), \end{aligned}$$

but the constant a is unknown. To estimate it, define a second state variable as $x_2(k) = a$ and derive the augmented non-linear system

$$\begin{aligned} x_1(k) &= x_2(k-1)x_1(k-1) + w_1(k-1), \\ x_2(k) &= x_2(k-1), \end{aligned}$$

and

$$z(k) = x_2(k)x_1(k) + v(k).$$

Figure 11.9 shows the results obtained by applying the EKF and the MC filter on simulated data from this model. In this case, EKF is doing a reasonable job. Even so, the MC filter (using $B = 1000$ and taking the means as estimates) again perform better. Note in particular how the estimate of a converge towards its true value

$a = 1.0$. The histograms in Figure 11.10 indicate that using the mean is sensible in this case also. □

11.5 Problems

Exercise 11.1 (EKF and the linearized filter)
Explain in detail why the EKF may be expected to work better than the linearized filter.

Exercise 11.2 (Random walk)
Consider again the random walk example, that is the model

$$\begin{aligned} x(k) =& x(k-1) + w(k-1) \\ z(k) =& x(k) + v(k) \end{aligned}$$

where $w(k-1) \sim \mathcal{N}(0, Q)$, and $v(k) \sim \mathcal{N}(0, R)$.

(a) Extend the state vector to $\mathbf{x}^*(k) = (x(k), \eta(k))^T$ and show that the system equations now can be written as

$$\begin{aligned} \mathbf{x}^*(k) =& \mathbf{\Phi}\mathbf{x}^*(k-1) + \varepsilon(k-1) \\ z(k) = \mathbf{H} * x(k) + v(k) \end{aligned}$$

where now $\varepsilon(k-1) \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_m)$, and $\mathbf{Q}_m$ is a 2×2 matrix with only non-zero element which is equal to one. Specify $\mathbf{\Phi}, \mathbf{Q}_m$ and $\mathbf{H}$.

(b) Run a Kalman filter on the extended model and show that the point estimates and the variances obtained for $x(k)$ are equal to those obtained by the original model. Use $Q = 1.0, R = 0.5$ and $m0 = 0.0, P0 = 1.0$.

Now our aim of extending the state vector is not to obtain point estimates in another way, but because the extended model is more suitable for estimation of parameters using the ideas discussed in this chapter. We will only consider estimation of Q, assuming R is known.

(c) Extend the model further by including $\sqrt{Q}$ in the state vector. Write down the system and observation equations in this case.

(d) Discuss why we have chosen to include Q in the model from (a) and not directly from the original model.

(e) Implement the extended Kalman filter on the model from (c) and use this to estimate Q.

(f) Now implement the Monte Carlo filter. Does this filter obtain better results than the extended Kalman filter?

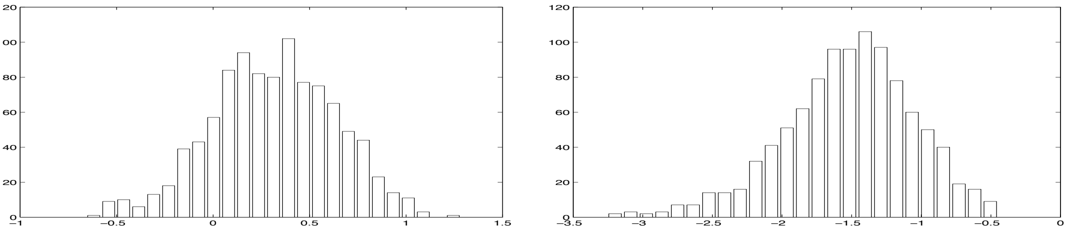

Figure 11.8: Direction measurements. Histogram of samples from $p(\mathbf{x}(25)|\vec{z}(25))$. Left panel shows $x_1(25)$ while right panel shows $x_2(25)$.

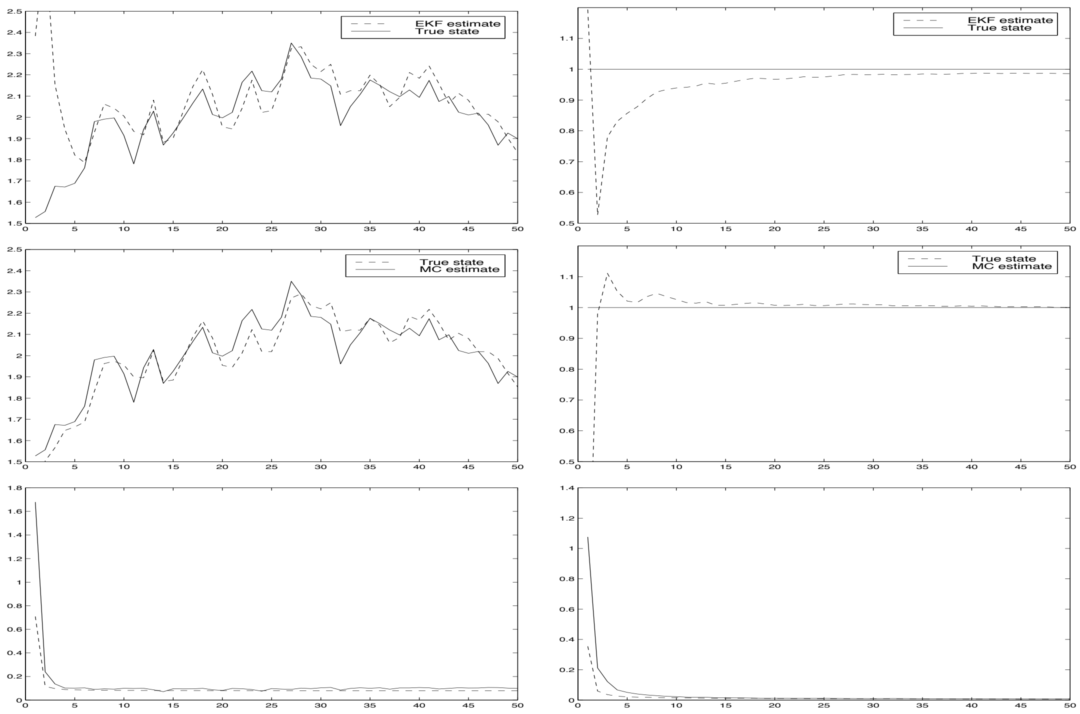

Figure 11.9: Parameter estimation. Left: State estimates and true states for $x(k)$. Right: State estimates and true states for a. The upper panels show the EKF estimates, the middle panels show the estimates from the MC filter, while the lower panel shows the variance estimates.

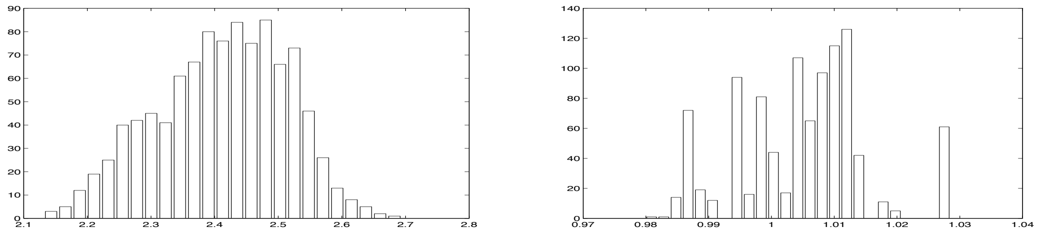

Figure 11.10: Parameter estimation. Histogram of samples from $p(\mathbf{x}(25)|\vec{z}(25))$. Left panel shows $x_1(25)$ while right panel shows $x_2(25)$.

Appendix A

Matlab commands

Properly used, *Matlab* is a superb experimental tool that allows us to explore the theory as well as getting suggestions for possible underlying theoretical relationships. In this chapter, we list some useful *Matlab* commands. For most of the examples, also included are the *Matlab* codes and functions which have been used for obtaining the different results. To get the details, type `help` followed by the function name in the *Matlab* window.

In *Matlab* , indices of vectors and matrices run from 1 and upwards. This cause some problems concerning the covariance matrices $\{\mathbf{Q}(k)\}$ of the state vector, since these run from 0 and upwards. In all the routines we therefore have used $\mathbf{Q}(k)$ as the covariance matrix for $\mathbf{w}(k-1)$ instead.

A.1 Properties of deterministic systems

In this section, we list some useful commands for the concepts discussed in Chapter 2.

Solving differential equations like (2.1) in *Matlab* can be performed by using the commands `ode23` or `ode45`, employing Runge-Kutta numerical integration methods of order 2/3 and 4/5, respectively. (One may also use the graphically based *SIMULINK* package within *Matlab* to solve differential (and difference) equations.)

For time-varying systems, the routines `lsim` and `dlsim` can be applied.

Matlab provides the routines `c2d` and `c2dm` (continuous - to - discrete) for transforming a continuous system to a discrete one. These routines compute the matrices $\mathbf{\Phi}$ and $\mathbf{\Psi}$ for time invariant systems with a fixed sampling time (cf. (2.7)).

In order to compute eigenvalues of a matrix, the command `eig` can be used. For calculating a matrix exponential, the routine `expm` is appropriate.

In order to check reachability, *Matlab* provides the special purpose routines `ctrb` and ctrbf. For observability, *Matlab* has special purpose routines obsv and obsvf.

A.2 *Matlab* commands for the Gaussian distribution

Some useful commands related to Gaussian distributions are

randn	Normally distributed random numbers (see also normrnd).
mvnrnd	Random matrices from the multivariate normal distribution.
normcdf	Normal cumulative distribution function (cdf).
normpdf	Normal probability density function (pdf).
normspec	Plots normal density between specification limits.

A.3 Code concerning the random walk example

The following *Matlab* commands were used for obtaining the variables in Figure 4.1:

```
randn('state',0);
n = 100;
Q = [ones(n/2,1); ones(n/2,1)/10];
R = ones(n,1)/2;
w = normrnd(zeros(n,1),sqrt(Q));
x = cumsum(w);
z = x+normrnd(zeros(n,1),sqrt(R));
a = 1:n;
plot(a,x,a,z,'*');
```

The following routine was used for using the Kalman filter on the data from Example 3:

```
function [xhat,P] = klm_rw(z,Q,R)
%Kalman filter for random walk model
%Assumes P(0) is infinity
[n,m] = size(z);
P = zeros(n,1);
xhat = zeros(n,1);
P(1) = R(1);
xhat(1) = z(1);
for k=2:n
  P1 = P(k-1)+Q(k);
  S = P1+R(k);
  K = P1/S;
  P(k) = (1-K)*P1;
  x1 = xhat(k-1);
  xhat(k) = x1+K*(z(k)-x1);
end
```

The following routine was used for calculating the fixed-point smoothers for the random walk process:

```
function [xhatt,Pt] = smo_point_rw(z,Q,R,t)
%Routine for estimating x(t) based on z(1),...,z(k)
[n,m] = size(z);
Pt = zeros(n,1);xhatt = zeros(n,1);
P = R(1);                 %k=1, using P(0) equal to infinity
xhat = z(1);
Pt(1) = P + sum(Q(2:t));xhatt(1) = xhat;
for k=2:t                 %Kalman filter up to t
  P1 = P+Q(k);S = P1+R(k);K = P1/S;P = (1-K)*P1;
  x1 = xhat;xhat = x1+K*(z(k)-x1);
  xhatt(k) = xhat;        %Prediction of x(t)
  if(k<t)                 %Prediction error of x(t)
    Pt(k) = P + sum(Q((k+1):t));
  else
    Pt(k) = P;
  end
end
P = [P,P;P,P];H = [0,1];  %Augmented model for k>t
xhat = [xhat;xhat];
for k=(t+1):n             %Kalman on augmented model
  P1 = P+[0,0;0,Q(k)];S = H*P1*H'+R(k);K = P1*H'./S;
  P = (diag([1,1])-K*H)*P1;
  x1 = xhat;xhat = x1+K*(z(k)-H*x1);
  xhatt(k) = xhat(1);
  Pt(k) = P(1,1);
end
```

For the fixed-interval smoother, only a small routine (utilizing the Kalman filter routine defined above) is needed:

```
function [xhatn,Pn] = smo_int_rw(z,Q,R)
%Function for fixed-inverval smoothing for the random walk model
[n,m] = size(z);
```

```
[xhat,P] = klm_rw(z,Q,R);           %Kalman filter
Pn= zeros(n,1);xhatn = zeros(n,1);
Pn(n) = P(n);xhatn(n) = xhat(n);    %k=n directly from Kalman
for k = (n-1):-1:1                  %Downwards recursion
  P1 = P(k)+Q(k+1);A = P(k)/P1;
  Pn(k) = P(k)+A*(Pn(k+1)-P1)*A;
  xhatn(k) = xhat(k)+A*(xhatn(k+1)-xhat(k));
end
```

Fixed-lag smoothing:

```
function [xhatL,PL] = smo_lag_rw(z,Q,R,L)
%Routine for estimating x(k) based on z(1),...,z(k+L)
[n,m] = size(z);
H = [1,zeros(1,L)];                  %Parameters in augmented model
Phi=[1,zeros(1,L);diag(ones(1,L)),zeros(L,1)];
Qm = zeros(L+1,L+1);                 %Qm(1,1) is set in each iteration
P = diag(10000*ones(L+1,1));         %Initial values
xhat = diag(zeros(L+1,1));
Qm(1,1) = Q(1);
PL = zeros(n,1);
xhatL = zeros(n,1);
for k=1:n                            %Kalman filter
  Qm(1,1) = Q(k);
  P1 = Phi*P*Phi'+Qm;S = H*P1*H'+R(k);K = P1*H'./S;
  P = (diag([ones(1,L+1)])-K*H)*P1;
  x1 = Phi*xhat;xhat = x1+K*(z(k)-H*x1);
  PL(k) = P(L+1,L+1);xhatL(k) = xhat(L+1);
end
xhatL = xhatL(L+1:n);PL = PL(L+1:n); %Shift
```

Calculation of (log-)likelihood.

```
function [loglik,Qhat,Rhat] = loglik_rw(z,q,r)
%Function for calculating log-likelihood for all values of Q and R
%given in the vectors q and r
%Also find the ML-estimates
[n,m] = size(z);
[Qm,Rm] = meshgrid(q,r);
konst = -0.5*n*log(2*pi);
loglik = zeros(size(Qm))+konst;
%Kalman filter for all Q and R values simultaneously
```

```
P = Rm;
xhat = ones(size(Qm)).*z(1);
for k=2:n
   P1 = P+Qm;
   S = P1+Rm;
   K = P1./S;
   P = (1-K).*P1;
   x1 = xhat;
   xhat = x1+K.*(z(k)-x1);
   loglik = loglik - 0.5.*log(S)-0.5.*(z(k)-x1).^2./S;
end
Qhat = Qm(loglik==max(max(loglik)));
Rhat = Rm(loglik==max(max(loglik)));
```

Note the use of matrix calculations, making it possible to calculate the (log-)likelihood for all sets of values for (Q, R) simultaneously.

Estimation by the EM-algorithm:

```
function [Qhat,Rhat] = em_rw(z,Q0,R0)
[n,m] = size(z);
s = 1;
Qhat = Q0;
Rhat = R0;
conv = 1;
while (conv>0.0001 & s < 1000)
  s = s+1;
  [xhatn2,Pn2] = smo_rw2(z,Qhat(s-1),Rhat(s-1));
  Qhat = [Qhat,mean(Pn2(2,2,2:n))+
         mean((xhatn2(2,2:n)-xhatn2(2,1:n-1)).^2)+
         mean(Pn2(2,2,1:n-1)-2*Pn2(1,2,1:n-1))];
  Rhat = [Rhat,mean((z'-xhatn2(2,:)).^2)+mean(Pn2(2,2,:))];
  conv = abs(Qhat(s)-Qhat(s-1))+abs(Rhat(s)-Rhat(s-1));
end
```

A.4 Code concerning the moving body example

A routine for simulating from the (discrete) moving body model (6.24):

```
function [x,z]= body(Phi,H,R,Q,P0,m0,N)
% Simulates the moving body example
```

```
%
% Phi : System matrix
% H: Measurement matrix
% R: Measurement error variance
% Q: System noise covariance matrix (assumed to be diagonal)
% P0: Initial error covariance matrix (assumed to be diagonal)
% m0: Imitial expected state estimate
% N:   Number of time steps to compute the solution
% x: Realization of the state vector - a (2,iter) matrix
% z: Realizations of the measurements - a (1,iter) matrix
%
x=zeros(2,N);
z=zeros(1,N);
x(:,1)= m0+[sqrt(P0(1,1))*randn(1,1);sqrt(P0(2,2))*randn(1,1)];
for k=2:N
x(:,k)=Phi*x(:,k-1)+[sqrt(Q(1,1))*randn(1,1);sqrt(Q(2,2))*randn(1,1)];
z(k)=H*x(:,k) + sqrt(R)*randn(1,1);
end
```

Kalman filtering of model (6.24):

```
function [xhat, xhat1,P,P1,K]= kfbody(z,Phi,H,R,Q,P0,m0,N)
%
% Generates Kalman filter estimates of the state vector of the
% moving body.
% The covariance matrix and Kalman gain also calculated.
%
% xhat:  xhat(k|k)
% xhat1: xhat(k|k-1)
% P:     P(k|k)
% P1:    P(k|k-1)
% K:     Kalman gain
%
% Initializing
xhat=zeros(2,N);
xhat1=zeros(2,N);
P=zeros(2,2,N);
P1=zeros(2,2,N);
K=zeros(2,N);
%
% k=1
P1(:,:,1)= Phi*P0*Phi' + Q;
```

```
S= H*P1(:,:,1)*H' + R;
K(:,1)= P1(:,:,1)*H'/S;
P(:,:,1)= (eye(2) - K(:,1)*H)*P1(:,:,1);
xhat1(:,1)= Phi*m0;
xhat(:,1)= xhat1(:,1) + K(:,1)*(z(1) - H*xhat1(:,1));
% Remaining k's
for k=2:N
 P1(:,:,k)= Phi*P(:,:,k-1)*Phi' + Q;
 S= H*P1(:,:,k)*H' + R;
 K(:,k)= P1(:,:,k)*H'/S;
 P(:,:,k)= (eye(2) - K(:,k)*H)*P1(:,:,k);
 xhat1(:,k)= Phi*xhat(:,k-1);
 xhat(:,k)= xhat1(:,k) + K(:,k)*(z(k) - H*xhat1(:,k));
end
```

Prediction:

```
function [xpred, Ppred]= kfbody_pred(z,Phi,H,R,Q,P0,m0,k,j)
%
% Make predictions of position at k+1,...,k+j
% for the moving body model
%
% xpred:  xhat(k+l|k)
% Ppred:  P(k+l|k)
%
% First run Kalman filter
[xhat, xhat1,P,P1,K]= kfbody(z,Phi,H,R,Q,P0,m0,k);
% Then run Kalman filter on modified model
Hm = [0,0];
xpred=zeros(2,k+j);
Ppred=zeros(2,2,k+j);
xpred(:,1:k)=xhat;
Ppred(:,:,1:k)=P;
%
% j=1
P1= Phi*P(:,:,k)*Phi' + Q;
S= Hm*P1*Hm' + R;
K= P1*Hm'/S;
```

```
Ppred(:,:,k+1)= (eye(2) - K*Hm)*P1;
xhat1= Phi*xhat(:,k);
xpred(:,k+1)= xhat1 + K*(0 - Hm*xhat1);
% Remaining l's
for l=2:j
 P1= Phi*Ppred(:,:,k+l-1)*Phi' + Q;
 S= Hm*P1*Hm' + R;
 K= P1*Hm'/S;
 Ppred(:,:,k+l)= (eye(2) - K*Hm)*P1;
 xhat1= Phi*xpred(:,k+l-1);
 xpred(:,k+l)= xhat1 + K*(0 - Hm*xhat1);
end
```

Kalman filtering with $\mathbf{K}(k) = 0$:

```
function [xhat, xhat1,P,P1,K]= kfbody_k0(z,Phi,H,R,Q,P0,m0,N)
%
% Generates Kalman filter estimates using K=0
% Initializing
xhat=zeros(2,N);
xhat1=zeros(2,N);
P=zeros(2,2,N);
P1=zeros(2,2,N);
K=zeros(2,N);
%
% k=1
P1(:,:,1)= Phi*P0*Phi' + Q;
S= H*P1(:,:,1)*H' + R;
K(:,1)= 0;
P(:,:,1)= (eye(2) - K(:,1)*H)*P1(:,:,1);
xhat1(:,1)= Phi*m0;
xhat(:,1)= xhat1(:,1) + K(:,1)*(z(1) - H*xhat1(:,1));
% Remaining k's
for k=2:N
 P1(:,:,k)= Phi*P(:,:,k-1)*Phi' + Q;
 S= H*P1(:,:,k)*H' + R;
 K(:,k)= 0;
 P(:,:,k)= (eye(2) - K(:,k)*H)*P1(:,:,k);
 xhat1(:,k)= Phi*xhat(:,k-1);
 xhat(:,k)= xhat1(:,k) + K(:,k)*(z(k) - H*xhat1(:,k));
end
```

A.5 Nonlinear filtering

A.5.1 Rejection sampling

A direct implementation of rejection sampling for the example in Section 11.3.1:

```
function [x,rej] = rej_samp(z,m,q,r,B)
x = zeros(B,1);
rej = zeros(B,1);
M = 1/(sqrt(2*pi*r));
for i=1:B
 x(i) = normrnd(m,sqrt(q));
 u = rand(1);
 p = normpdf(z,x(i)^2,sqrt(r))/M;
 rej(i) = 0;
 while(u>p)
  rej(i) = rej(i)+1;
  x(i) = normrnd(m,sqrt(q));
  u = rand(1);
  p = normpdf(z,x(i)^2,sqrt(r))/M;
 end
end
```

For loops through all the samples may be inefficient in *Matlab* . An alternative is to generate all samples simultaneously by utilizing the vector ability of *Matlab* . In particular, at the first step B samples are generated. At the next only those samples that not were accepted are sampled again, and this continue untill all samples are generated. The code is as follows:

```
function x = rej_samp_fast(z,m,q,r,B)
M = 1/(sqrt(2*pi*r));
%Sample for all B
x = normrnd(m,sqrt(q),B,1);
```

```
u = rand(B,1);
p = normpdf(z,x.*x,sqrt(r))/M;
acc = u <= p;
samp = B;
ind = 1:B;
while(sum(acc)<B)
 ind2 = ind(acc==0);
 n_a = length(ind2);
 samp = samp+n_a;
 x(ind2) = normrnd(m,sqrt(q),n_a,1);
 u(ind2) = rand(n_a,1);
 p = normpdf(z,x.*x,sqrt(r))/M;
 acc = u <= p;
end
```

Simulating data for Example 5:

```
function [x,z] = sim_gss(Q,R,m0,P0,N);
%Generating sample from Gordon, Salmond & Smith example
x0 = normrnd(m0,sqrt(P0));
x = zeros(N,1);
z = zeros(N,1);
x(1) = 0.5*x0+25*x0/(1+x0^2)+8*cos(1.2*0)+normrnd(0,sqrt(Q));
z(1) = x(1)^2/20+normrnd(0,sqrt(R));
for k=2:N
 x(k) = 0.5*x(k-1)+25*x(k-1)/(1+x(k-1)^2)+
        8*cos(1.2*(k-1))+normrnd(0,sqrt(Q));
 z(k) = x(k)^2/20+normrnd(0,sqrt(R));
end
```

Extended Kalman filtering on Example 5:

```
function [xhat,P] = ekf_gss(z,Q,R,m0,P0)
%Extended Kalman filter for Gordon, Salmond & Smith example
[n,m] = size(z);
P = zeros(n,1);
xhat = zeros(n,1);
%
Phi = 0.5+(25-25*m0^2)/(1+m0^2)^2;
x1 = 0.5*m0+25*m0/(1+m0^2)+8*cos(1.2*0);
H = x1/10;
P1 = Phi*P0*Phi+Q;
```

```
S = H*P1*H+R;
K = P1*H/S;
P(1) = (1-K*H)*P1;
xhat(1) = x1+K*(z(1)-x1^2/20);
for k=2:n
  Phi = 0.5+(25-25*xhat(k-1)^2)/(1+xhat(k-1)^2)^2;
  x1 = 0.5*xhat(k-1)+25*xhat(k-1)/(1+xhat(k-1)^2)+8*cos(1.2*(k-1));
  H = x1/10;
  P1 = Phi*P(k-1)*Phi+Q;
  S = H*P1*H+R;
  K = P1*H/S;
  P(k) = (1-K*H)*P1;
  xhat(k) = x1+K*(z(k)-x1^2/20);
end
```

Monte Carlo filter on Example 5:

```
function [xhat,P,samp] = mcfilt_gss(z,Q,R,m0,P0,B)
%MC filter filter for Gordon, Salmond & Smith example
%Initialization
[N,m] = size(z);
P = zeros(N,1);
xhat = zeros(N,1);
samp = zeros(N,1);
%k=0
xstar = normrnd(m0,sqrt(P0),B,1);
ind = 1:B;
for k=1:N
 %Making prediction distribution
 x0 = xstar;
 xmean = mean(x0);
 xvar = var(x0);
 lambda=0.2;
 kon = sqrt((1-lambda^2)*B/(B-1));
 x0= xmean+kon*(xstar-xmean);
 %Sampling x(k) from pred. distr
 b = unidrnd(B,B,1);
 x0star = x0(b)+normrnd(0,lambda*sqrt(xvar),B,1);
 xstar = 0.5.*x0star+25.*x0star./(1+x0star.*x0star)+
         8.*cos(1.2.*(k-1))+normrnd(0,sqrt(Q),B,1);
 p = normpdf(z(k),xstar.*xstar/20,sqrt(R));
 u = rand(B,1);
 acc = u <= p;
 samp(k) = B;
 %Rejection sampling (fast implementation)
 while(sum(acc)<B)
  ind2 = ind(acc==0);
  n_a = length(ind2);
  samp(k) = samp(k)+n_a;
  b = unidrnd(B,n_a,1);
  x0star = xmean+kon*(x0(b)-xmean)+normrnd(0,lambda,n_a,1);
  xstar(ind2) = 0.5.*x0star+25.*x0star./(1+x0star.*x0star)+
                8.*cos(1.2.*(k-1))+normrnd(0,sqrt(Q),n_a,1);
  u(ind2) = rand(n_a,1);
  p = normpdf(z(k),xstar.*xstar/20,sqrt(R));
  acc = u <= p;
  if samp(k)>30000
    k
    sum(acc)
  end
 end
 k
 samp(k)
 xhat(k)=mean(xstar);
 P(k)=var(xstar);
end
```

Bibliography

[1] B. D. O. Anderson and J. B. Moore. *Optimal filtering*. Electrical Engineering Series. Prentice Hall, 1979.

[2] W. E. Boyce and R. C. DiPrima. *Elementary Differential Equations and Boundary Value Problems*. John Wiley & Sons, 1977.

[3] R. G. Brown and P. Y. C. Hwang. *Introduction to random signals and applied Kalman filtering*. John Wiley, 1992.

[4] R. S. Bucy and K. D. Senne. Digital synthesis of non-linear filters. *Automatica*, 7:287–298, 1971.

[5] N. Gordon, D. Salmond, and A. F. M. Smith. Novel approach to nonlinear/non-Gaussian Bayesian state estimation. *IEE Proceedings-F*, 140(2):107–113, 1993.

[6] M.S. Grewal and A.P. Andrews. *Kalman filtering - Theory and practice*. Prentice Hall, 1993.

[7] P. J. Harrison and C. F. Stevens. Bayesian forecasting. *Journal of Royal Statistical Society, Series B*, 38:205–247, 1976.

[8] A. C. Harvey. *Forecasting, structural time series models and the Kalman filter*. Cambridge University Press, 1989.

[9] R. E. Kalman. A new approach to linear filtering and prediction problem. *ASME Journal of Basic Engineering, series D*, 82:34–45, 1960.

[10] G. Kitagawa. Non-Gaussian state-space modeling of non-stationary time series. *JASA*, 82(400):1032–1041, 1987.

[11] K. V. Mardia, J. T. Kent, and J. M. Bibby. *Multivariate Analysis*. Academic Press, 1979.

[12] G. Minkler and J. Minkler. *Theory and application of Kalman filtering*. Magellan Book Company, 1992.

[13] H. H. Taylor and S. Karlin. *An introduction to Stochastic modelling.* Academic Press, second edition, 1993.

[14] M. West and J. Harrison. *Bayesian forecasting and dynamic models.* Springer Series in Statistics. Springer-Verlag, New York, second edition, 1989.

Index